Organizing Structured Web Sources by Query Schemas: A Clustering Approach*

Bin He, Tao Tao, Kevin Chen-Chuan Chang
Computer Science Department
University of Illinois at Urbana-Champaign
{binhe, taotao}@uiuc.edu, kcchang@cs.uiuc.edu

ABSTRACT

In the recent years, the Web has been rapidly "deepened" with the prevalence of databases online. On this deep Web, many sources are *structured* by providing structured query interfaces and results. Organizing such structured sources into a domain hierarchy is one of the critical steps toward the integration of heterogeneous Web sources. We observe that, for structured Web sources, query schemas (*i.e.*, attributes in query interfaces) are discriminative representatives of the sources and thus can be exploited for source characterization. In particular, by viewing query schemas as a type of categorical data, we abstract the problem of source organization into the clustering of categorical data. Our approach hypothesizes that "homogeneous sources" are characterized by the same hidden generative models for their schemas. To find clusters governed by such statistical distributions, we propose a new objective function, *model-differentiation*, which employs principled hypothesis testing to maximize statistical heterogeneity among clusters. Our evaluation over hundreds of real sources indicates that (1) the schema-based clustering accurately organizes sources by object domains (*e.g.*, Books, Movies), and (2) on clustering Web query schemas, the model-differentiation function outperforms existing ones, such as likelihood, entropy, and context linkages, with the hierarchical agglomerative clustering algorithm.

Categories and Subject Descriptors

H.2.5 [**Database Management**]: Heterogeneous Databases; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Measurement

Keywords

data integration, deep Web, hierarchical agglomerative clustering

*This material is based upon work partially supported by NSF Grants IIS-0133199 and IIS-0313260. 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.

CIKM'04, November 8–13, 2004, Washington, DC, USA.

1. INTRODUCTION

In the recent years, the Web has been rapidly "deepened" with the prevalence of databases online. As Figure 1 conceptually illustrates, on this so-called "deep Web" (*i.e.*, Web sources backed by databases), numerous online databases provide dynamic *query*-based data access through their *query interfaces*, instead of static URL links. A July 2000 survey [3] estimated that 96,000 "search cites" and 550 billion content pages in the deep Web. Our recent study [7] in April 2004 estimated 450,000 online databases.

This paper focuses on *structured* databases on the deep Web, which provide structured query interfaces and return structured objects (*e.g.*, a `Book` source like *amazon.com* queries and returns books with author, title; a `Car` source like *cars.com* queries and returns cars with make, model). The deep Web provides much structured information, by presenting data from back-end relational DBMSs. Our survey [7] distinguished such structured sources from unstructured *text* sources (of documents)– Not surprisingly, structured sources dominated by a 3:1 ratio.

The structured deep Web thus presents challenges for *large-scale* information integration: While there are myriad useful databases, how can a user *find* the right sources and *query* them in a right way? Consider user Amy, who is moving to a new town. First, different queries need different sources to answer: Where can she look for real estate listings? (*e.g.*, *realtor.com.*) Studying for a new car? (*cars.com.*) Looking for a job? (*monster.com.*) Further, different sources support different query capabilities: After source hunting, Amy must then learn the grueling details of querying.

While tantalized by the need for effectively accessing the deep Web, the integration over large-scale structured sources has largely remained unexplored. As Section 2 will discuss, on one hand, research on large scale distributed search has mostly focused on the integration over *text* sources. On the other hand, the efforts of database integration have so far assumed relatively small scale, pre-configured sources– in contrast to dynamic source selection. With structured sources proliferating (and dominating) on the Web, their integration is increasingly more critical.

As a first step toward such integration, this paper studies organizing sources by "structures"– We propose to characterize sources by their *query schemas*, *i.e.*, attributes in their query interfaces, *e.g.*, in Figure 1: Q_{az} = {author, ..., publisher} for *amazon.com*; {city, ..., rent} for *apartments.com.* Specifically, for structured Web sources, we believe query schemas are right "representatives" for structured sources: First, they are readily *available*, on the "surface" of online databases, and thus can be easily "crawled" (unlike sophisticated querying required for accessing data inside). Second, they are *discriminative*: The query schema of a source characterizes its *object domain* (*e.g.*, Books, Movies) and *query capabilities* (*e.g.*, by author, director), both of which are essential for distinguishing structured sources. In particular, given Web pages containing query

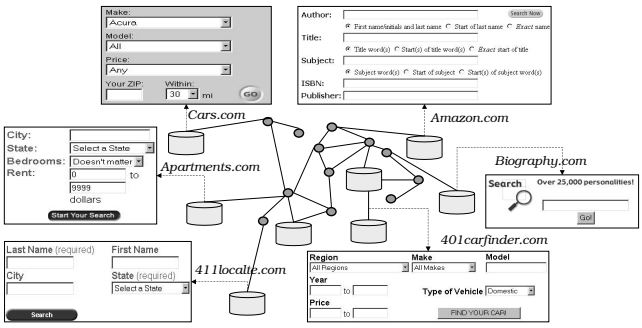


Figure 1: Query-based data sources on the Web.

interfaces in HTML format, we have solved the extraction of query schemas from interfaces with a parsing approach [32].

Schema-based characterization will thus organize sources by their object domains and query capabilities, which is preliminary for supporting both *source selection* (by finding sources of the same object domains) and *query mediation* (by finding sources with similar query attributes). Given a set of query schemas representing structured sources, our task is thus to construct a *domain hierarchy*, where each object domain contains a set of "structurally-homogeneous" sources.

In abstraction, this problem is essentially a clustering problem– Our goal is to cluster structured Web source into their domain hierarchy. In particular, by viewing a schema (*e.g.*, Q_{az}) as a *transaction* (*e.g.*, with items author, . . . , publisher) and thus a special type of *categorical data*, we abstract our problem of source organization as the clustering of a set of categorical data. As such data are typically sparse in a high-dimensional space, conventional clustering based on measuring the similarities between pairwise data points (*e.g.*, the Jaccard coefficient [19]) does not work well. Several recent efforts have thus developed new *objective functions*, *e.g.*, context-linkages [13] and entropy [8].

Our framework pursues probabilistic model-based clustering with a new objective function. To begin with, motivated by our real-world observations (Section 3), we hypothesize that homogeneous sources share the same hidden generative model, which probabilistically generates schemas from a finite vocabulary of attributes. This hypothesis naturally matches model-based clustering– to form clusters from different models. Further, to realize such clustering, we propose a new objective function: *model-differentiation* or MD, which seeks to maximize *statistical heterogeneity* among clusters. Rather than relying on ad-hoc cluster-similarity measures, MD takes principled hypothesis testing in statistics, called *test of homogeneity* [5], to evaluate if multiple clusters of data are generated from homogeneous distributions.

Specifically, we develop Algorithm MD_{hac} for clustering query schemas to build a domain hierarchy. First, we assume the statistical model of a cluster is a *multinomial distribution* over attributes observed in the cluster. This modeling greatly simplifies the homogeneity testing, thus enables MD-driven clustering. Second, guided by the MD objective, we adopt χ^2 testing for evaluating the homogeneity among clusters. Third, for hierarchy construction, we use the general HAC (hierarchical agglomerative clustering) approach [2, 10, 22].

However, while hypothesis testing is appealing as a principled cluster-similarity measure, it is challenging to apply in clustering: To start, initial clusters must be formed with a minimal size to warrant "sufficient observations" (as required by a statistical method). Given individual schemas as input, we "bootstrap" by *pre-clustering* them into some initial clusters, from which hypothesis testing can take over to construct final clusters by HAC. Further, for those initial clusters that are too small to continue, we handle them (after HAC clustering) by *post-classifying* each "loner-schema" to the constructed clusters. We develop Algorithm MD_{hac} with such pre-clustering and post-classification techniques, making hypothesis testing possible in clustering.

We experimented with about 500 real sources in 8 domains (*e.g.*, Airfares, Automobiles, Books). Our goals are two-fold: (1) to evaluate the effectiveness of schema-based clustering for organizing structured sources into a domain hierarchy, and (2) to evaluate the performance of the MD objective function by comparing to the existing approaches using context linkages, log-likelihood, and entropy. The results show the effectiveness in both aspects (Section 6).

In summary, the contributions of this paper are:

- We propose to organize structured Web sources by their query schemas and further abstract the problem as the clustering of categorical data.
- We propose model-differentiation as a new objective function for clustering, which allows principled statistical measure for determining cluster homogeneity. Our evaluation shows that, on clustering the Web query schemas, the model-differentiation function outperforms existing ones.

We review related work in Section 2 and Section 3 presents our motivating observations of structured sources on the Web. Section 4 discusses our statistical cluster modeling and the MD objective function. Section 5 develops Algorithm MD_{hac}. Section 6 reports our experiments and Section 7 concludes this paper.

2. RELATED WORK

We relate our work to the literature in three aspects: in terms of the integration scenario, the Web clustering problem and our clustering technique.

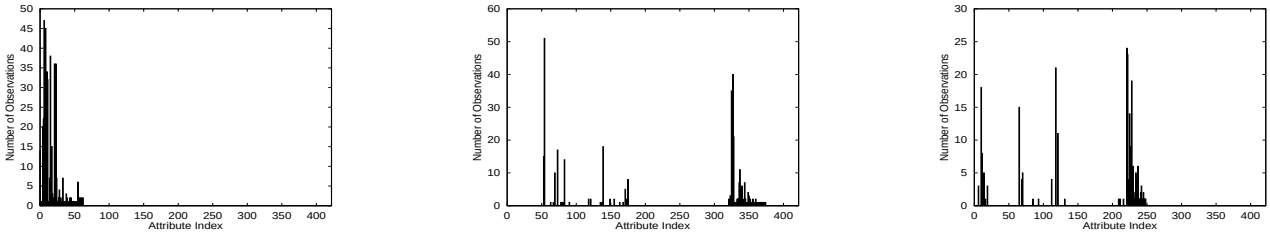
First, in terms of the *integration scenario*, this paper studies clustering structured sources on the Web. Our goal of clustering sources to facilitate large-scale integration (Section 1) has largely been unexplored. On one hand, for structured (relational) sources, *information integration* has mainly assumed relatively small-scaled, pre-configured systems [28, 9] (*e.g.*, Information Manifold [21], TSIMMIS [26]). On the other hand, research efforts on large scale distributed search has mostly focused on *text* sources, *e.g.*, source characterization [6], source classification [25, 18], and query routing [23]. Our focus mixes both of the above: We aim to enable *large-scale* integration over *structured* databases– which has not been extensively studied before. We thus propose to organize these sources by their query schemas, which characterize both their object domains and query capabilities. Such schema-based characterization will likely help to give a meaningful organization to the myriad Web sources.

In particular, the result of this work can support other integration tasks such as query mediation and source selection. For instance, our schema matching work [14, 15] studied discovering semantically corresponding attributes across Web interfaces in the same domain (*e.g.*, Book sources). To find such "homogeneous" sources (that are meaningful to integrate), this paper thus provides the preliminary step of source organization.

Second, in terms of the *Web clustering problem*, existing Web clustering works mainly focus on clustering Web documents by exploiting Web content and linkage information [30, 31, 29, 17]. In contrast, this paper focuses on the clustering of structured Web sources. With the observation that query schemas are discriminative representatives of sources, we are able to translate the original problem of source organization into the clustering of query schemas, a type of categorical data.

domain	*number of sources*	*domain*	*number of sources*
Airfares	53	Hotels	38
Automobiles	102	Jobs	55
Books	69	Movies	78
CarRentals	24	MusicRecords	75

Figure 2: Our dataset of sample deep web sources: 494 sources in 8 domains.



(a) Airfares. **(b)** Movies. **(c)** Hotels.

Figure 3: Attribute frequencies of different domains.

Third, in terms of the *clustering technique*, this paper proposes a model-differentiation objective function for clustering query-schema data. Clustering of general categorical data has recently been more actively studied, *e.g.*, STIRR [12], CACTUS [11], ROCK [13], and COOLCAT [8]. STIRR treats clustering as a partitioning problem of hypergraph and solves it based on non-linear dynamical systems. CACTUS considers a cluster as a set of pairwise strong connected attributes by measuring attribute occurrences. ROCK, COOLCAT and this paper pursue the same direction of defining a new similarity measure involving the *global context* (such as properties of a entire cluster) instead of local pairwise measure. ROCK uses context linkages between data points, and COOLCAT uses entropy of clusters. As an alternative, we develop the model-differentiation measure, which maximizes the statistical heterogeneity among clusters. Section 6 compares these related approaches.

Our statistical approach belongs to the general idea of model-based clustering (*e.g.*, partitional EM algorithm [24] and hierarchical algorithms [2, 10]). Such clustering assumes that data is generated from a mixture of distributions, each of which defines a cluster. This general approach is traditionally not specific to categorical data– More recently, reference [22] proposes a multivariate multinomial distribution (in which each feature is an independent multinomial distribution) for categorical data. In comparison, the model we propose for schema data (or transactional data) is a "joint" multinomial, where all features are generated from a multinomial distribution (Section 4.1).

All the existing model-based works essentially use likelihood as the objective function to maximize– In contrast, we propose model-differentiation (Section 4.2) by maximizing the statistical heterogeneity among clusters. In our extended report [16], we show that these two objective functions are in fact *equivalent* in assessing the global clustering results. However, toward their "global" objectives, they indeed imply *different* greedy "local" similarity measures (which Section 4.3 will develop). In our experiments, we also compare the model-differentiation measure with the likelihood one on HAC algorithm.

3. MOTIVATION

With virtually unlimited amount of information, the deep Web is clearly an important frontier for data integration. To characterize this "wild" frontier, we extensively studied sources on the deep Web in our recent survey [7], which this section is based upon. (Some observations here were also reported in our earlier work [14].) We collected a dataset of deep Web sources using Web directories (*e.g.*, InvisibleWeb.com, BrightPlanet.com, Web-File.com) and search engines (*e.g.*, Google.com). As Figure 2 summarizes, the collected dataset consists of 494 sources in 8 domains (*e.g.*, Airfares, Automobiles, etc.). Our experiments (Section 6) use the same dataset. This section reports two observations pertinent to our focus of source organization– In fact, these observations motivated our approach.

First, we observe that query schemas are *discriminative* representatives of structured sources. Specifically, we count attribute frequencies for each domain (*i.e.*, the aggregate occurrences of an attribute across all sources in the same domain). Figure 3 lists the attribute frequencies (y-axis) of 3 domains (Airfares, Hotels and Movies) over all the attributes (x-axis) in the 8 domains. We observe that each domain contains a dominant range of attributes, distinctive from other domains. For example, Airfares only covers the first 53 attributes and does not overlap with Movies. Hotels has its dominant range of attributes from index 200 to 250 (while overlapping with Airfares in some of the first 53 attributes).

Further, some attributes are only observed in one domain– These *anchor attributes* make their domains more distinguishable. For instance, make and model are anchor attributes for Automobiles, and ISBN for Books. We observed that most schemas indeed contain anchor attributes. In particular, our dataset indicates that 457 out of the total 494 interfaces, accounting for 92.5%, contain some anchor attributes. The prevalence of anchor attributes motivates our bootstrapping techniques (Section 5.1). We believe that the existence of anchor attributes might not be a unique phenomenon for schema data– For other types of transactional data, it is likely that a cluster will contain some *anchor items* that are characteristics of the cluster.

Second, we observe that the aggregate schema vocabulary of sources in the same domain tends to converge at a relatively small size with respect to the growth of sources. Figure 4 shows, for each domain, the growth of vocabularies as sources increase in numbers. The curves clearly indicate the convergence of vocabularies. Since the vocabulary growth rates (*i.e.*, the slopes of these curves) decrease rapidly, as sources proliferate, their vocabularies will tend to stabilize. This observation indicates that homogeneous sources (in the same domain) share some *concerted* vocabulary of attributes.

These two observations together motivate our approach: The first "discriminative" observation suggests using query schemas as "representatives" of sources in the source organization, which is essentially a clustering problem. Our goal is thus to cluster structured Web source into their domain hierarchy. By viewing a schema as a transaction and thus a special type of categorical data, we abstract our problem of source organization as the clustering of cate-

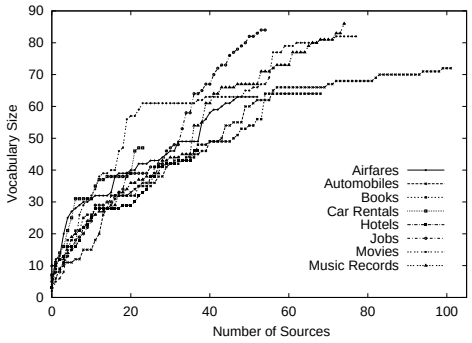


Figure 4: Schema vocabularies.

gorical data. Further, the second "concerted" observation leads us to hypothesize the existence of a hidden schema model (for each domain), which generates the observed query schemas. We thus pursue model-based clustering (Section 4). Finally, the "discriminative" observation further hints a novel objective function, model-differentiation, which seeks to maximize statistical heterogeneity among models in clustering.

4. MD-BASED CLUSTERING

As just abstracted and motivated, we are pursuing a MD-based approach to cluster query schemas. In the literature, model-based clustering has been widely discussed. The general idea can be stated as: The population of interest consists of G clusters, generated by G different models. Given a set of data points (a set of schemas) $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_n}\}$, where each $\mathbf{x_i}$ is independently generated from one of the G models, $\mathcal{M}_1$,...,$\mathcal{M}_G$, the probability of generating $\mathbf{x_i}$ in the kth model is $Pr(\mathbf{x_i}|\mathcal{M}_k)$. A clustering of $\mathbf{X}$ is a partition of $\mathbf{X}$ into G groups: denoted by $(\mathbf{X}; P) = (C_1, ..., C_G)$, where P partitions $\mathbf{X}$. The objective of model-based clustering is to identify the partition P that all $\mathbf{x_i}$ generated from the same model $Pr(\bullet|\mathcal{M}_k)$ are partitioned into a single group.

To realize this general model-based clustering of query schemas, we design a model as a multinomial distribution (Section 4.1) and develop model-differentiation as the new *objective function* of clustering based on statistical hypothesis testing. Specifically, guided by this objective function, we adopt the commonly used χ^2 testing. (Section 4.2). Unlike the clustering work in statistic software, which also use χ^2 testing, we apply it for categorical data based on the generative model. Since we are pursuing a hierarchical clustering approach, we apply the widely used HAC (hierarchial agglomerative clustering) algorithm, which needs a measure to quantify the "similarity" between two clusters. In particular, we derive a new similarity measure from the MD objective function (Section 4.3).

4.1 Hypothesis Modeling

To develop the MD-based clustering, we need to define the generative model. To begin with, we first introduce our model definition as multinomial distribution. Specifically, we assume attributes are independent each other, which is a commonly used assumption for text data [27]. Then we describe how a model generates a schema in a statistical way and further how to generate a cluster of schemas.

First, to define the model for the task of schema clustering, we need to describe what is a schema. We view a query schema as a set of attributes for a query interface, as Section 1 introduced. (Figure 1 illustrates some real query interfaces.) For instance, for *amazon.com*, the query schema Q_{az} is {author, ..., publisher} (as shown in Section 1). For simplicity, in later examples, we denote attributes in letters A, B,....

Our first attempt is to consider a schema as a set of *distinct* attributes. Therefore, we view the generation of a schema as *sampling without replacement* [5] from a set of attributes, which means the result of a trial (to select an attribute) is not the same as any previous trials. (The trials are therefore "stateful".) That is, we can consider a schema with n attributes as an experiment with n trials; once one attribute is selected, it will not be selected again in the subsequent trials. However, while this model is accurate, its "stateful" trials result in complicated homogeneity testing.

Our second attempt is to approximate the generation process by *sampling with replacement* [5], where the attributes can be repeatedly selected in a schema. With this alternative strategy, to generate a schema Q in some cluster C, the model $\mathcal{M}$ behind C is a *multinomial model* with parameters p_1,...,p_N. More specifically, a multinomial model $\mathcal{M}$ for C consists of an exhaustive set of N mutually exclusive events (In our problem, the events are in fact the attributes.) A_1,...,A_N (which covers all the attributes observed in C) with associated probabilities p_1,...,p_N, $\sum_{j=1}^{N} p_j = 1$. We denote $\mathcal{M}$ as $\mathcal{M} = \{A_1{:}p_1,...,A_N{:}p_N\}$. Each trial of $\mathcal{M}$ generates one of the N events. The probability of generating an attribute A from $\mathcal{M}$ in a single trial is

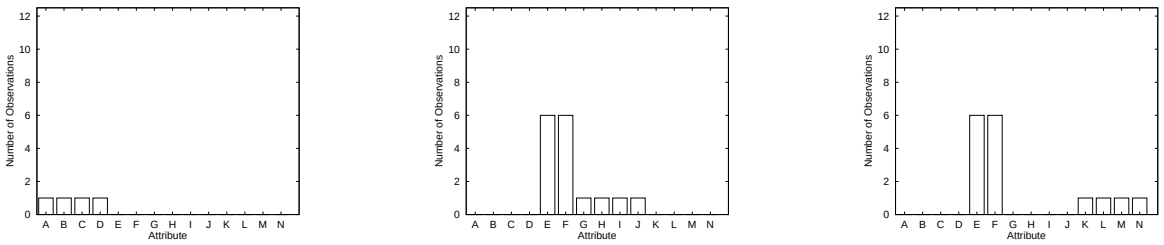
$$Pr(\mathsf{A}|\mathcal{M}) = \begin{cases} p_i, & \exists i : \mathsf{A} = A_i \\ 0, & otherwise \end{cases} \quad (1)$$

Next, we discuss the generation of a schema in cluster C from $\mathcal{M}$. Under this multinomial model, a schema Q is characterized by its observed attributes (and their frequencies). We thus view Q (of length n) as $Q = \{A_1{:}y_1,...,A_N{:}y_k\}$, $\sum_{i=1}^{N} y_i = n$, where y_i is the frequency (number of occurrences) of attribute A_i in Q. For a schema with distinct attributes, y_i is either 0 or 1. For instance, for the query schema Q_{az} of *amazon.com*, the frequency of author, y_{author}, is 1. (We discuss later that this model can generate schemas with duplicate attributes.) That is, by definition of standard multinoimal distribution [5], Q (of length n) is generated from $\mathcal{M}$ as the result of n independent (therefore "stateless") trials with the following probability:

$$Pr(Q|\mathcal{M}, n) = n! \prod_{i=1}^{N} \frac{Pr(A_i|\mathcal{M})^{y_i}}{y_i!}. \quad (2)$$

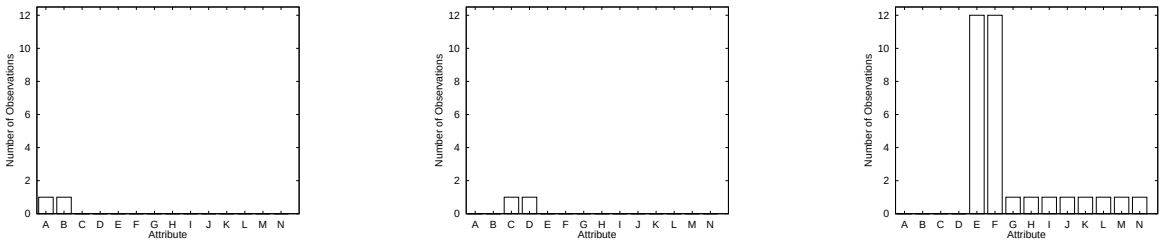
Example 1: Consider a cluster C with 4 schemas: Q_1:{A,B,C}, Q_2:{A,B}, Q_3:{C,D}, and Q_4:{C,D,E}. The model $\mathcal{M}$ contains 5 attributes (events): A, B, C, D and E, with probabilities p_1, p_2, p_3, p_4 and p_5 respectively. Under our multinomial modeling, we view a schema as a set of attribute frequencies (*i.e.*, y_{ji}). For example, Q_1 = {A:1, B:1, C:1, D:0, E:0}. In particular, $y_{11} = 1$ since A occurs once in Q_1. The probability of generating Q_1 is $Pr(Q_1|\mathcal{M}, 3) = 6p_1p_2p_3$. ■

Then, we discuss how we statistically view a cluster of schemas. Consider a cluster of schemas $C = \{Q_1, Q_2, ..., Q_m\}$, where each schema Q_j (with length n_j) is generated by the same model $\mathcal{M}$ = $\{A_1{:}p_1,...,A_N{:}p_N\}$. Since each Q_j is a multinomial experiment of n_j trials, we can view C as an experiment with $\sum_{j=1}^{m} n_j$ trials by concatenating the trials in all schemas. That is, we consider that C is a series of sampling from the same multinomial distribution $\mathcal{M}$ (*i.e.*, the same p_1,...,p_N), with all these independent trials. The theoretical explanation is as follows: Let all Q_j = $\{A_1{:}\mathbf{y_{j1}},...,A_N{:}\mathbf{y_{jN}}\}$, where $\mathbf{y_{ji}}$'s are random variables denoting the frequencies of A_i, share the same multinomial distribution $\mathcal{M}$ = $\{A_1{:}p_1,...,A_N{:}p_N\}$. For the entire C, we define new random variables $\mathbf{z_1}$,...,$\mathbf{z_N}$ as aggregate attribute frequencies. That



(a1) $C_{<1,2>}$. **(a2)** C_3. **(a3)** C_4.

(a) Multinomial distributions of the clustering result by combining C_1 and C_2.



(b1) C_1. **(b2)** C_2. **(b3)** $C_{<3,4>}$.

(b) Multinomial distributions of the clustering result by combining C_3 and C_4.

Figure 5: Comparison of two possible clustering results.

is, $\mathbf{z_i} = \sum_{j=1}^{m} \mathbf{y_{ji}}$. In our extended report [16], we show that $\mathbf{z_1}, ..., \mathbf{z_N}$ are also sampled from the same multinomial distribution $\mathcal{M}$ with $\sum_{j=1}^{m} n_j$ trials. Therefore, under this multinomial view, we can express C as aggregate attribute frequencies, *i.e.*, $C = \{A_1{:}z_1, ..., A_N{:}z_N\}$.

Example 2: Continue on the cluster C in Example 1, by considering C is generated by a multinomial distribution and computing z_i as $\sum_{j=1}^{m} y_{ji}$, we can express cluster C as {A:2, B:2, C:3, D:2, E:1}. For example, A has frequencies 2 because it occurs once in both Q_1 and Q_2. ■

The simple multinomial modeling simplifies hypothesis homogeneity testing (by directly fitting the contingency table as shown in Section 4.2). However, the modeling is inaccurate: It may generate some schemas that are not observable in the real world. For instance, it may generate a schema {author, author, title}, where author is repeated twice. While the modeling seems crude (like other typical "independent" assumptions in, say, Naive Bayes Classifier for text), our empirical study shows that the simple model performs well.

As a remark, this modeling is much simpler than what we define in our previous work MGS [14]. The MGS work addresses matching schemas across sources in the same domain. (Therefore, this paper is a preliminary step to provide input for MGS.) The MGS modeling assumes a two-level model structure to capture concepts and synonyms for the goal of synonym discovery. This paper assumes a much simpler model because it is sufficient to capture the attribute frequencies across different domains for the purpose of clustering.

In this section, we develop the generative model. Next, we introduce the new objective function, model-differentiation, for clustering schema data and present the χ^2 testing to realize the MD function.

4.2 Model-Differentiation: A New Objective Function

Clustering must be guided by some *objective function* that specifies the property of the ideal clusters. Regardless of the objective function, the basic idea of clustering is to put similar data together and dissimilar data apart. For model-based clustering, similar data might be generated from the same underlying model, while dissimilar data from different models. Thus, we achieve better clustering result when the underlying models are more distinguishable.

	A_1	A_2	A_3	...	A_n	sum
C_1	O_{11}	O_{12}	O_{13}	...	O_{1n}	X_1
C_2	O_{21}	O_{22}	O_{23}	...	O_{2n}	X_2
...	...	...	...	...	...	...
C_m	O_{m1}	O_{m2}	O_{m3}	...	O_{mn}	X_m
sum	Y_1	Y_2	Y_3	...	Y_n	S

Figure 6: Contingency table for testing.

Example 3: As a running example, assume we are given four clusters of schemas, referred to as dataset $\mathcal{I}$: C_1:{A:1, B:1}, C_2:{C:1, D:1}, C_3:{E:6, F:6, G:1, H:1, I:1, J:1} and C_4:{E:6, F:6, K:1, L:1, M:1, N:1}. Now assume we want to generate 3 clusters (G = 3). To reduce the number of clusters to three, we need to combine two clusters into one. We denote the combination of clusters C_k and C_l as $C_{<k,l>}$.

We compare two possible clustering results, as illustrated in Figure 5. The first result (Figure 5(a)) combines C_1 and C_2, while the second result (Figure 5(b)) combines C_3 and C_4. Figure 5(a) is not as good as Figure 5(b) because the distributions of C_3 (Figure 5(a2)) and C_4 (Figure 5(a3)) are similar (and hence the schemas generated from these two models will also be similar). Figure 5(b) is better because the attributes with non-zero frequencies in the three clusters do not overlap. ■

Therefore, we define the objective function of clustering as some function $\mathcal{H}$ that characterizes the heterogeneity of models under a partition P, denoted by $\mathcal{H}(\mathbf{X}; P)$. The goal of clustering is to find the partition P that maximizes function $\mathcal{H}$, *i.e.*, $\arg\max_P \mathcal{H}(\mathbf{X}; P)$. In statistics, the homogeneity of distributions can be measured by *test of homogeneity* using statistical hypothesis testing. More specifically, if we have a partition function P partitioning $\mathbf{X}$ into clusters $C_k (1 \leq k \leq G)$, we can test the hypothesis "$C_k (1 \leq k \leq G)$ are sampled from the same distribution" with standard testing approaches. The result of testing is a probabilistic variable λ to indicate the confidence that we accept the hypothesis that those distri-

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	sum
$C_{<1,2>}$	1	1	1	1	0	0	0	0	0	0	0	0	0	0	4
C_3	0	0	0	0	6	6	1	1	1	1	0	0	0	0	16
C_4	0	0	0	0	6	6	0	0	0	0	1	1	1	1	16
sum	1	1	1	1	12	12	1	1	1	1	1	1	1	1	36

Figure 7: Example of model-differentiation testing.

butions are the same. Thus the heterogeneity of models is $1 - \lambda$. Formally, the MD-based clustering is to find

$$\begin{aligned} \arg\max_P \mathcal{H}(\mathbf{X}; P) &= \arg\max_P \mathcal{H}(C_1, ..., C_G) \\ &= \arg\max_P \{1 - \lambda(C_1, ..., C_G)\} \\ &= \arg\min_P \lambda(C_1, ..., C_G), \end{aligned} \quad (3)$$

where $\lambda(C_1, ..., C_G)$ is the result of hypothesis testing on a partition P with G clusters.

More specifically, given a partition P on the observed data $\mathbf{X}$, we apply χ^2 hypothesis testing to compute $\lambda(C_1, ..., C_G)$. In statistics, χ^2 testing can be used to test the homogeneity among multiple clusters with multinomial distributions by constructing a *contingency table*. Since we show that a cluster of schemas is also generated by a multinomial distribution, we can directly apply the test of homogeneity by fitting the attribute frequencies in the cluster into the contingency table, which reflects the fact that our modeling simplifies the testing. For arbitrary models, it deserves further research efforts to figure out how to fit them into the contingency table.

Formally, assume there are m clusters $C_1, ..., C_m$, and each of them is generated from its own multinomial distribution (as defined in Section 4.1). There are n different events (attributes) altogether, denoted by $A_1, ..., A_n$. Figure 6 is the contingency table to show this set of data. In particular, O_{ij} stands for the attribute frequency of A_j in cluster C_i. X_i is the sum of all the O_{ij} in ith row and Y_j is the sum of all the O_{ij} in jth column. That is, $X_i = \sum_{j=1}^{n} O_{ij}$ and $Y_j = \sum_{i=1}^{m} O_{ij}$. S is the sum of all O_{ij} in the table. Thus $S = \sum_{i=1}^{m} X_i = \sum_{j=1}^{n} Y_j$.

We want to test the hypothesis: $\forall j, 1 \leq j \leq n$, $p_{j1} = p_{j2} = ... = p_{jm} = \frac{Y_j}{S}$, where p_{ji} is the probability of observing attribute A_j in cluster C_i. This hypothesis is tested by considering the random variable

$$D^2(C_1, ..., C_m) = \sum_{i=1}^{m} \sum_{j=1}^{n} [\frac{(O_{ij} - X_i \times \frac{Y_j}{S})^2}{X_i \times \frac{Y_j}{S}}]. \quad (4)$$

It can be shown that D^2 has asymptotically a χ^2 distribution with $(n-1)(m-1)$ degree of freedom, denoted by *df* [1].

We have to use both D^2 and *df* to decide how similar the m clusters are. D^2 value itself is not a valid indicator for the similarity of clusters without being qualified the degree of freedom. Therefore we need to translate these two values into a single similarity measure. In statistics, we can compute the P-value given D^2 and *df*, denoted by $PV(D^2, df)$. The P-value is the probability value λ in Equation 3, indicating the confidence that we accept the hypothesis that the m clusters are generated from the same distribution. The objective function $\mathcal{H}$ is then

$$\mathcal{H}(C_1, ..., C_G) = 1 - PV(D^2, df). \quad (5)$$

The computation of P-value is expensive and requires numerical integration. Therefore, in practice, we develop an alternative measure, $\tilde{\mathcal{H}}$, by applying a normalized D^2 value. In particular, to make the D^2 values of different degrees of freedom (resulted from different clusters) comparable, we use the D^2 values with a commonly adopted significance level 0.5% as the normalization factors, denoted by $D_s^2(df)$, with different degrees of freedom. We consider $\tilde{\mathcal{H}}$ the ratio between the computed D^2 value and the D_s^2 with the same *df*:

$$\tilde{\mathcal{H}}(C_1, ..., C_G) = \frac{D^2}{D_s^2(df)}. \quad (6)$$

Example 4: Consider the first clustering result in Example 3, we want to test the hypothesis that these three clusters are generated from the same distribution. The corresponding contingency table of this scenario is listed in Figure 7. Applying Equation 4, we get $D^2(C_{<1,2>}, C_3, C_4) = 34.33$ and $df(C_{<1,2>}, C_3, C_4) = (14 - 1) \times (2 - 1) = 13$. the D_s^2 value for 0.5% with $df = 13$ is 29.82. By applying Equation 6, $\mathcal{H}(C_{<1,2>}, C_3, C_4) = \frac{34.33}{29.82} = 1.15$.

Consider the second clustering result in Example 3 similarly, we get $D^2(C_1, C_2, C_{<3,4>}) = 65.67$ and $df(C_1, C_2, C_{<3,4>}) = (14 - 1) \times (2 - 1) = 13$. We then have $\mathcal{H}(C_1, C_2, C_{<3,4>}) = \frac{65.67}{29.82} = 2.2 > \mathcal{H}(C_{<1,2>}, C_3, C_4)$, which means the second clustering result is better than the first one. ■

Require: SchemaSet $\mathbf{X}$, ObjectiveFunction $\mathcal{F}$, NumberOfClusters G
1: **/* Form a list of initial V clusters */**
2: $C_k = X_k$, $(1 \leq k \leq V)$
3: **/* Derive similarity measure */**
4: s = a similarity measure derived from $\mathcal{F}$
5: **/* HAC main framework */**
6: **for** $K = V, V-1, ..., G$ **do**
7: **/* Compute pairwise similarities */**
8: $k^*, l^* = \arg\min_{k,l} s(C_k, C_l)$, $(1 \leq k < l \leq K)$
9: **/* Merge the most similar two clusters*/**
10: $C_{<k^*,l^*>} = \text{MERGE}(C_{k^*}, C_{l^*})$
11: **end for**

Figure 8: General HAC algorithm GE_{hac}.

4.3 General HAC Algorithm And MD-Based Similarity Measure

For constructing the domain hierarchy as motivated in Section 1, we adopt the general HAC clustering approach, which is widely used for data clustering [20]. Figure 8 illustrates the general HAC framework [22]. In HAC, we need to measure the similarity of clusters. That is, given a set of clusters, C_1,...,C_V, we compute all the pairwise values $s(k, l)$, where s is a similarity function from the objective function of clustering. The criterion of defining similarity function $s(k, l)$ is to maximize the objective function in each step. The two clusters with the smallest $s(k, l)$ are merged in each iteration. The algorithm stops when there are G clusters left.

Specifically, for our MD-based clustering, we derive $s(k, l)$ from $\mathcal{H}(\mathbf{X}; P)$ (defined in Section 4.2) as follows: In each iteration of HAC, we merge the clusters with the smallest $\mathcal{H}$ value (*i.e.*, the most similar two models) and therefore we define $s(k, l)$ to be

$$s(k, l) = \mathcal{H}(C_k, C_l). \quad (7)$$

Require: SchemaSet $\mathbf{X}$, NumberOfClusters G
1: **/* Form the initial clusters */**
2: $\mathcal{C}$ = DATAGROUPING($\mathbf{X}$)
3: **/* Move loner interfaces into $\mathcal{N}$ */**
4: $\mathcal{C}, \mathcal{N}$ = GROUPSELECTION($\mathcal{C}$)
5: **/* Standard HAC clustering with new measure */**
6: $\mathcal{C}$ = CLUSTERINGHAC($G, \mathcal{C}$)
7: **/* Classify loners into accomplished clusters */**
8: $\mathcal{C}$ = LONERHANDLING($\mathcal{N}, \mathcal{C}$)
9: **/* Build the domain hierarchy with HAC approach */**
10: BUILDHIERARCHY($\mathcal{C}$)

Figure 9: Algorithm MD_{hac}.

Example 5: Consider the dataset $\mathcal{I}$ in Example 3 as the input of GE_{hac}. Assume we want to generate 3 clusters. We compute all the pairwise similarities with Equation 7 and get $s(1,2) = 0.43$, $s(1,3) = 0.96$, $s(1,4) = 0.96$, $s(2,3) = 0.96$, $s(2,3) = 0.96$, and $s(3,4) = 0.04$. It is clearly to see that C_3 and C_4 are most similar. Thus the clustering result is C_1, C_2 and $C_{<3,4>}$. ■

5. CLUSTERING QUERY SCHEMAS: ALGORITHM MD_{HAC}

In this section, we present the concrete algorithm MD_{hac} (denoting MD-based HAC algorithm) by solving the difficulty of applying the MD-based clustering. To test the heterogeneity of models with hypothesis testing (Section 4.2), we have to face one challenge: When the observations of events are not sufficiently large, the value of D^2 may not be closely converged to χ^2 distribution and thus affects the value of $\mathcal{H}$. In particular, the χ^2 test requires each event (attribute in our case) has at least 5 observations to ensure the approximation of χ^2 distribution [1]. However, the input data are initially collected without being grouped and thus cannot satisfy this requirement.

To address this problem of insufficient observations, we design pre-clustering and post-classification techniques. Pre-clustering is to pre-cluster the data into groups with sufficient observations to satisfy the requirement of hypothesis testing. Post-classification is to classify the insufficient *loner* schemas excluded by bootstrapping into the accomplished clusters.

In our development, pre-clustering consists of two steps: data grouping and group selection. Data grouping is to merge the data into groups by using deterministic rules. After grouping, some groups contain sufficient observations, while others not. Group selection only selects those groups with sufficient observations to participate in the HAC clustering. We consider the insufficiently observed schemas as loner schemas. Post-classification is essentially the classification of loners into the completed clusters, which we call loner handling in our implementation.

Figure 9 shows Algorithm MD_{hac}: First, DATAGROUPING pre-clusters data into groups based on the corollaries developed from the existence of anchor attributes (Section 5.1). Second, GROUPSELECTION excludes the loner schemas with loner threshold N (Section 5.2). Third, CLUSTERINGHAC clusters the remaining groups with the standard HAC algorithm and Equation 7 as the similarity measure. Fourth, LONERHANDLING classifies the loner schemas into the accomplished G clusters (Section 5.3). Finally, BUILDHIERARCHY again applies the HAC algorithm to build the hierarchical tree of domains (by considering each cluster as one domain).

5.1 Data Grouping

Our pre-clustering technique leverages the existence of *anchor attributes* to group schemas deterministically. Our exploration for the schemas of the 8 domains indicates that most schemas contain anchor attributes (Section 3). Specifically, an anchor attribute is essentially an attribute with non-zero probability only for one cluster. More formally,

Definition 1: Given a clustering partition $C_1, ..., C_G$ and assume the model under C_k is $\mathcal{M}_k$, an attribute A is an *anchor attribute* if there is only one C_k that contains A, *i.e.*, $Pr(\mathsf{A}|\mathcal{M}_k) > 0$ and $Pr(\mathsf{A}|\mathcal{M}_l) = 0$ for $l \neq k$. A schema is a *distinguishable schema* if it contains at least one anchor attribute. ■

Definition 1 implies the following corollaries:

Corollary 1: If A is an anchor attribute with $Pr(\mathsf{A}|\mathcal{M}_k) > 0$ and A is observed in a schema Q with length n, then $Q \in C_k$, $Pr(Q|\mathcal{M}_k, n) > 0$ and $Pr(Q|\mathcal{M}_l, n) = 0$ for any $l \neq k$. ■

PROOF. Assume $Q = \{A_1{:}y_1,...,A_s{:}y_s\}$ of length n and $\mathsf{A} = A_t$, $y_t > 0$ since A is observed in Q. By applying Equation 2, we have $Pr(Q|\mathcal{M}_l, n) = n! \prod_{i=1}^{s} \frac{Pr(A_i|\mathcal{M}_l)^{y_i}}{y_i!}$. For $l \neq k$, $Pr(\mathsf{A}|M_l) = 0$ according to definition of anchor attribute, thus we have $Pr(Q|\mathcal{M}_l, n) = 0$. Since Q must belong to some cluster, Q has to be clustered into C_k, thus $Q \in C_k$ and $Pr(Q|\mathcal{M}_k, n) > 0$.

Corollary 2: If Q_1 is a distinguishable schema and $Q_1 \subseteq Q_2$, Q_2 is also a distinguishable schema and belongs to the same cluster as Q_1. ■

PROOF. Assume $Q_1 \in C_k$. Q_2 must belong to some cluster C_l. If $l \neq k$, Q_1 becomes a schema containing overlapping attributes of C_k and C_l. Thus $Pr(Q_1|\mathcal{M}_l, n) > 0$, which contradicts the assumption that Q_1 is a distinguishable schema. Therefore we have $l = k$, which means Q_2 is in the same cluster as Q_1.

Corollary 2 indicates that if all the schemas are distinguishable schemas, the containment relation is correct in grouping data. Guided by Corollary 2, we group the query schemas by putting all the schemas satisfying Corollary 2 into one cluster. More specifically, we first randomly select a schema Q, and put all the query schemas Q_i satisfying $Q \subseteq Q_i$ or $Q_i \subseteq Q$ into the same bucket of Q. We then evaluate all the Q_i just added recursively until no satisfied schema can be found. It can be shown that the output of data grouping is not affected by the random selection of schemas.

However, Corollary 2 requires that the schemas are distinguishable schemas. Since it is difficult to affirm whether a schema is distinguishable before clustering, we design a heuristic by observing the difference of the *containing set* of distinguishable and indistinguishable schemas. We define the containing set of a schema Q, denoted by $S(Q)$, as all the Q_is satisfying $Q \subseteq Q_i$ in the dataset. Intuitively, for a distinguishable schema Q, the schemas in $S(Q)$ are in one domain (based on Corollay 2) and hence they should be more overlapping in attributes; While for an indistinguishable schema Q, the schemas in $S(Q)$ come from multiple domains and they should be more different in attributes. Hence, we design a step of *schema type checking* before grouping: For each schema Q, we compute its containing set $S(Q)$. Then for any Q_i and Q_j in $S(Q)$, we compute their distance $d(i,j)$ as $\frac{|Q_i \cap Q_j|}{|Q_i \cup Q_j|}$. If there exists $d(i,j) < \theta$, where θ is a threshold value, we consider Q an indistinguishable schema and exclude it to participate in data grouping. (In fact, the excluded schemas effectively become loner schemas in group selection). In our experiment, we set $\theta = 0.2$. We assume the remaining schemas are all distinguishable schemas and apply Corollary 2 to group them.

Example 6: Consider a set of 8 schemas: Q_1:{C}, Q_2:{A,B}, Q_3:{A,B,C,E}, Q_4:{A,D}, Q_5:{A,B,D,E}, Q_6:{C,F}, Q_7:{C,F,G}, and Q_8:{C,H}. First, we do the schema type checking on every schema. In particular, Q_1's containing set $S(Q_1)$=$\{Q_3, Q_6, Q_7, Q_8\}$.

Computing the pairwise distance of schemas in $S(Q_1)$, we know the minimal distance is $d(3,7) = 1/6 < 0.2$. Therefore, Q_1 is indistinguishable schema and excluded for grouping. Similarly, we check other schemas and they all pass this checking.

Next, we start to group the remaining schemas by randomly choosing a schema, say Q_2. Then we find $Q_2 \subseteq Q_3$ and $Q_2 \subseteq Q_5$. By recursively evaluating Q_3 and Q_5, we find $Q_4 \subseteq Q_5$ and no more schemas can be incorporated. Therefore Q_2, Q_3, Q_4 and Q_5 are in one group. We repeat this process on the remaining schemas and find Q_6 and Q_7 are in another group and Q_8 itself is in the third group. The excluded Q_1 is considered as an individual group. Hence, data grouping outputs four groups.

Without schema type checking, the data grouping will output only one group with all schemas together since Q_1 only contains an overlapping attribute C, which is observed in all the groups. ■

5.2 Group Selection

While data grouping merge the data into groups, some groups may still have insufficient observations, which may affect the result of hypothesis testing. Therefore, we consider those groups as loner groups, not participating in the CLUSTERINGHAC step in Algorithm MD_{hac}. The criterion to judge loner groups is to set a *loner threshold* N: If the frequencies of all attributes in a group are lower than N, we consider it as a loner group and all the schemas in this group as loners. In statistics, N is conventionally set to 5, which is the recommended value for χ^2 hypothesis testing [1]. In our experiment, we find setting N to 3 is enough to contain sufficient observations.

Example 7: Consider the four groups in Example 6, the multinoimal expressions of these four groups (Q_2, Q_3, Q_4, Q_5), (Q_6, Q_7), (Q_1) and (Q_8) are: (A:4, B:4, C:1, D:2, E:1), (C:2, F:2, G:1), (C:1) and (C:1, H:1) respectively. If we set the threshold N to 2, then groups (Q_1) and (Q_8) are considered as loner groups. ■

5.3 Loner Handling

After the step of CLUSTERINGHAC, we classify the loners into the accomplished clusters. As a classification problem, we classify a loner schema Q into the cluster with the largest probability to observe it. Formally, given a schema Q of length n, we will classify Q into the cluster C_i with the highest $Pr(Q|\mathcal{M}_i, n)$.

Some loners may have zero probabilities for all clusters. Equation 1 shows that when an attribute A_j does not exist in a cluster C_i, $Pr(A_j|\mathcal{M}_i) = 0$. For a schema Q with attributes not in any cluster, all the probabilities $Pr(Q|\mathcal{M}_i, n)$ will be 0 and thus we cannot decide which cluster to classify it into. To avoid this problem, in this step, we set $Pr(A_j|\mathcal{M}_i)$ to a very small value ε instead of 0 if A_j is not observed in C_i. In our implementation, we set $\varepsilon = 10^{-3}$.

Example 8: Continue with Example 7, assume after HAC clustering, the two clusters cannot be merged. We name group (Q_2, Q_3, Q_4, Q_5) as C_1 and (Q_6, Q_7) as C_2. From Section 4.1, we know multinomial distribution of C_1 is (A:0.33, B:0.33, C:0.08, D:0.17, E:0.08) and of C_2 is (C:0.4, F:0.4, G:0.2).

Now we need to classify loners Q_1 and Q_8 into these two clusters. For Q_1, by applying Equation 2, we have $Pr(Q_1|\mathcal{M}_1, 1)$ = 0.08 and $Pr(Q_1|\mathcal{M}_2, 1)$ = 0.4. Therefore, Q_1 is put into cluster C_2. Similarly, Q_8 is also put into cluster C_2. The final result of this clustering is (Q_2, Q_3, Q_4, Q_5) and (Q_1, Q_6, Q_7, Q_8). ■

5.4 Time Complexity

We evaluate the time complexity of MD_{hac}, for each individual step. Assume we have n schemas in G clusters with totally m attributes. Also, we assume the longest length of one schema is a constant C. DATAGROUPING can be executed in $O(C^2n^2)$ = $O(n^2)$ time since we need to compare one schema with all the remaining schemas to check the containment relationship in Corollary 2. GROUPSELECTION can be executed in $O(mn)$ time in that for each group, we need to check the attribute frequencies. CLUSTERINGHAC takes $O(n^2m)$ time because every time we combine two clusters C_k and C_l, we only need to recompute the similarities between the remaining clusters and the new cluster $C_{<k,l>}$. The similarities between other clusters are not changed. So each iteration takes $O(nm)$ time and there are at most n iterations. Hence, we have the $O(n^2m)$ upper bound for this step. LONERHANDLING takes $O(nmG)$ time because for each loner schema, we need to check G clusters with the computation of probability over at most m attributes. The final step BUILDHIERARCHY is similar to CLUSTERINGHAC and takes $O(G^2m)$ time. Therefore, the time complexity altogether is bounded by $O(n^2m)$.

6. EXPERIMENTS

To evaluate the MD_{hac} algorithm, we test it with 8 domains of structured sources on the deep Web. We compare our model-differentiation based approach with likelihood [22], entropy (COOLCAT) [8] and context linkage (ROCK) [13] based approaches using HAC algorithm and analyze the results. Also, we show the domain hierarchy built by MD_{hac} and evaluate the influence of the loner threshold N on the clustering performance.

6.1 Experiment Setup

We collected 494 sources over 8 domains, totally covering 422 attributes (Section 3). For each source, we manually extract attributes from its query interface by extracting noun phrases, and then judge its corresponding domain. This is our ground truth of "correct" clustering. The reason we do not apply our work in [32] for interface extraction is that we want to isolate the clustering process to study and thus fairly evaluate the performance.

To measure the result of clustering, we adopt the *conditional entropy* introduced in [4]. For a given number of clusters G, the value of the conditional entropy is within the range from 0 to $\log G$, where 0 denotes the 100% correct clustering, $\log G$ denotes purely random clustering result, *i.e.*, the sources from every single domain are evenly distributed into all clusters. Thus, the closer the conditional entropy value is to 0, the better the result is.

6.2 Experimental Results

We design three suites of experiments. *First*, we compare our approach MD_{hac} with the three existing approaches: likelihood based approach (LK_{hac}), entropy based approach (EP_{hac}) and context linkage based approach (CL_{hac}) for clustering the sources of 8 domains. For fair comparison, we only replace the similarity measure of test of model difference (Equation 6) in the CLUSTERINGHAC step with the likelihood based measure, entropy based measure and context linkage based measure. All the rest settings (pre-clustering and post-classification etc.) stay the same and the loner threshold N is set to 3.

To make the other measures clear, we briefly list each of them below. Reference [22] introduces the likelihood based similarity measure for HAC algorithm as Equation 8. The basic idea is that in each merging step in HAC, the two clusters generating the maximal likelihood after merging will be merged.

$$s(k,l) = \mathcal{L}(C_k) + \mathcal{L}(C_l) - \mathcal{L}(C_{<k,l>}). \quad (8)$$

COOLCAT [8] introduces entropy as the objective function, from where we derive the following similarity measure for HAC algorithm, with the same idea as the derivation of Equation 8 in [22].

MD_{hac}	Af	Am	Bk	Cr	Ht	Jb	Mv	Mr
C_1	0	101	0	0	2	4	0	0
C_2	0	0	62	0	0	1	9	2
C_3	0	0	0	24	0	0	0	0
C_4	0	0	0	0	35	0	0	1
C_5	0	0	0	0	0	50	1	0
C_6	53	0	0	0	1	0	0	0
C_7	0	0	0	0	0	0	8	67
C_8	0	1	7	0	0	0	62	7

(a) Conditional entropy of MD_{hac}: 0.32.

LK_{hac}	Af	Am	Bk	Cr	Ht	Jb	Mv	Mr
C_1	0	100	0	0	2	8	0	0
C_2	0	0	62	0	0	1	7	2
C_3	0	0	0	0	35	6	0	1
C_4	0	0	0	0	0	0	56	5
C_5	0	2	7	0	0	0	10	2
C_6	0	0	0	0	0	0	7	67
C_7	53	0	0	0	1	0	0	0
C_8	0	0	0	24	0	40	0	0

(b) Conditional entropy of LK_{hac}: 0.42.

EP_{hac}	Af	Am	Bk	Cr	Ht	Jb	Mv	Mr
C_1	0	100	0	0	2	4	0	0
C_2	0	0	62	0	0	0	5	2
C_3	0	0	0	24	0	0	0	0
C_4	0	0	0	0	35	6	0	1
C_5	0	0	0	0	0	0	57	5
C_6	0	0	0	0	0	0	8	67
C_7	53	0	0	0	1	0	0	0
C_8	0	2	7	0	0	45	10	2

(c) Conditional entropy of EP_{hac}: 0.38.

CL_{hac}	Af	Am	Bk	Cr	Ht	Jb	Mv	Mr
C_1	34	0	0	0	1	0	0	0
C_2	19	0	0	0	0	0	0	0
C_3	0	99	7	0	2	7	1	1
C_4	0	1	62	24	0	1	4	1
C_5	0	0	0	0	35	21	0	1
C_6	0	0	0	0	0	26	1	0
C_7	0	0	0	0	0	0	70	42
C_8	0	2	0	0	0	0	4	32

(d) Conditional entropy of CL_{hac}: 0.61.

Figure 10: Comparison of four similarity measures in HAC.

$$s(k,l) = |C_k|E(C_k) + |C_l|E(C_l) - |C_{<k,l>}|E(C_{<k,l>}). \quad (9)$$

ROCK [13] introduces context linkage as the similarity measure:

$$s(k,l) = \frac{link[C_k, C_l]}{(n_k + n_l)^{(1+2f(\theta))} - n_k^{(1+2f(\theta))} - n_j^{(1+2f(\theta))}}. \quad (10)$$

The result in Figure 10 shows the comparison of the four measures in HAC algorithm. In particular, we present the results as the numbers of Web sources in each cluster from each domain. For example, in Figure 10 (a), 101 stands for that there are, in cluster C_1, 101 Web sources from automobile domain. We use the abbreviations Af, Am, Bk, Cr, Ht, Jb, Mv and Mr to denote the 8 domains Airfares, Automobiles, CarRentals, Hotels, Jobs, Movies and MusicRecords respectively. Figure 10 illustrates two results: 1) It is feasible to address the clustering of structured sources as the clustering of query schemas. The matrix of MD_{hac}, LK_{hac} and EP_{hac} do show correct clustering for most data. The result of (CL_{hac}) is not good perhaps because its similarity measure may not fit the schema data well; 2) MD_{hac} achieves, on clustering Web schemas, the best performance (smallest conditional entropy) among all the measures. In particular, compared with the second best measure, EP_{hac}, MD_{hac} has better clustering results for Jobs and Movies.

Second, we show the effectiveness of MD_{hac} to build the domain hierarchy (as Section 1 motivated). After clustering 8 domains, we continue with the BUILDHIERARCHY step to build the domain hierarchy in the same way as the HAC clustering. The result in Figure 11 illustrates that Automobiles and Jobs are merged in the same subtree, MusicRecords, Books and Movies in another subtree, and Airfares, CarRentals and Hotels in a third subtree. This hierarchy is consistent with our observation in the real world (*i.e.*, object domains are characterized by their query schemas): Books, MusicRecords and Movies are all media and often sold together online, and so are Airfares, CarRentals and Hotel reservations. Automobiles and Jobs are together because they share many location information, such as city, state and zip code.

Finally, we design experiments to evaluate the influence of the loner threshold N. In statistics, 5 is the recommended for the accuracy of χ^2 hypothesis testing and therefore N does not need to be larger than 5. We let N range from 2 to 5 and test all the four

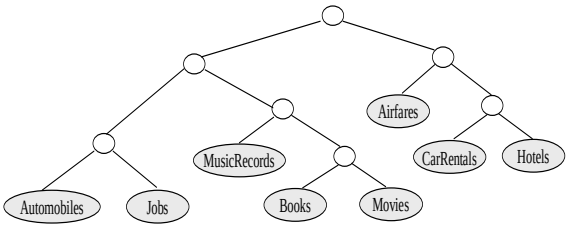


Figure 11: The domain hierarchy built by MD_{hac}.

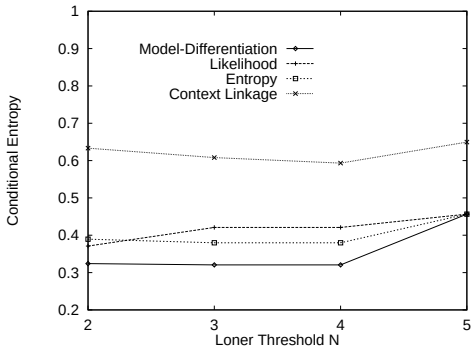


Figure 12: The influence of loner threshold N.

measures (We exclude $N = 1$ because $N = 1$ means no group selection). The result in Figure 12 shows that the clustering result is not affected too much by N, when N is ranged from 2 to 4. When $N = 5$, the result is worse because of the limited sampling size of our dataset. Setting N to 5 will trim most groups in group selection, where some insufficiently observed domains (*e.g.*, CarRental) are entirely trimmed out. Hence, we expect the result of $N = 5$ will be good when we have more observations. Putting in other words, when we have sufficient observations, the setting of N will not affect the result significantly.

7. CONCLUSION

This paper studies the problem of organizing structured sources on the Web, a task we believe is critical for large-scale integration of such sources. Motivated by our observations of the deep Web,

we propose to organize sources by their query schemas, and further abstract the problem as the clustering of categorical data. We develop a new model-differentiation objective function for clustering. Guided by the MD objective, we derive a new similarity measure for the general HAC algorithm. To apply statistical hypothesis testing for clustering, we design pre-clustering and post-classification techniques. Our experiments show the effectiveness of our abstraction– By clustering the query schemas, we can accurately organize sources into object domains. Also, we show that the model-differentiation function outperforms existing ones with the hierarchical agglomerative clustering algorithm.

8. REFERENCES

[1] A. Agresti. *Categorical Data Analysis*. John Wiley & Sons, Inc. New Jersey, 2002.

[2] J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49(3):803–821, 1993.

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

[4] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[5] H. D. Brunk. *An Introduction to Mathematical Statistics*. Blaisdell Pub. Co, 1965.

[6] J. P. Callan, M. Connell, and A. Du. Automatic discovery of language models for text databases. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 479–490, Philadelphia, Pennsylvania, USA, June 1999. ACM Press.

[7] K. C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang. Structured databases on the web: Observations and implications. *SIGMOD Record*, 33(3), 2004

[8] B. D., C. J., and L. Y. Coolcat: An entropy-based algorithm for categorical clustering. In *11th International Conference on Information and Knowledge Management*, pages 582–589, 2002.

[9] D. Florescu, A. Y. Levy, and A. O. Mendelzon. Database techniques for the world-wide web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.

[10] C. Fraley. Algorithms for model-based Gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, 1999.

[11] V. Ganti, J. Gehrke, and R. Ramakrishnan. CACTUS - clustering categorical data using summaries. In *Knowledge Discovery and Data Mining*, pages 73–83, 1999.

[12] D. Gibson, J. M. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB Journal*, 8(3–4):222–236, 1998.

[13] S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[14] B. He and K. C.-C. Chang. Statistical schema matching across web query interfaces. In *Proceedings of the 2003 ACM SIGMOD Conference (SIGMOD 2003)*, 2003.

[15] B. He, K. C.-C. Chang, and J. Han. Discovering complex matchings across web query interfaces: A correlation mining approach. In *Proceedings of the 2004 ACM SIGKDD Conference (SIGKDD 2004)*, 2004.

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

[17] X. He, H. Zha, C. Ding, and H. Simon. Web document clustering using hyperlink structures. Technical Report CSE-01-006, Dept. of Computer Science and Engineering, Pennsylvania State University, 2001.

[18] P. Ipeirotis and L. Gravano. Distributed search over the hidden-web: Hierarchical database sampling and selection. In *VLDB Conference*, 2002.

[19] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall Inc., Englewood Cliffs, NJ, 1988.

[20] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[21] A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of the 22nd VLDB Conference*, pages 251–262, Bombay, India, 1996. VLDB Endowment, Saratoga, Calif.

[22] M. Meilă and D. Heckerman. An experimental comparison of several clustering and initialization methods. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 386–395, 1998.

[23] W. Meng, K.-L. Liu, C. T. Yu, X. Wang, Y. Chang, and N. Rishe. Determining text databases to search in the internet. In *Proceedings of 24th International Conference on Very Large Data Bases*, pages 14–25, New York City, New York, USA, Aug. 1998. Morgan Kaufmann.

[24] T. M. Mitchell. *Machine Learning*. MIT Press,McGraw-Hill, 1997.

[25] M. S. Panagiotis G. Ipeirotis, Luis Gravano. Probe, count, and classify: Categorizing hidden web databases. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Santa Barbara, Ca., May 2001.

[26] Y. Papakonstantinou, H. García-Molina, and J. Ullman. Medmaker: A mediation system based on declarative specifications. In *Proceedings of the 12th International Conference on Data Engineering*, New Orleans, La., 1996.

[27] J. Ponte and W. Croft. A language modelling approach to information retrieval. In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.

[28] J. D. Ullman. Information integration using logical views. In *Proceedings of the 6th International Conference on Database Theory*, Delphi, Greece, Jan. 1997. Springer, Berlin.

[29] Y. Wang and M. Kitsuregawa. Evaluating contents-link coupled web page clustering for web search results. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, 2002.

[30] O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In *SIGIR Conference*, pages 46–54, 1998.

[31] O. Zamir, O. Etzioni, O. Madani, and R. M. Karp. Fast and intuitive clustering of web documents. In *Knowledge Discovery and Data Mining*, pages 287–290, 1997.

[32] Z. Zhang, B. He, and K. C.-C. Chang. Understanding web query interfaces: Best-effort parsing with hidden syntax. In *Proceedings of the 2004 ACM SIGMOD Conference (SIGMOD 2004)*, 2004.