Schema Matching Using Directed Graph Matching

[1]K.AMSHAKALA
Department of Computer Science Engineering and Information Technology
Coimbatore Institute of Technology, Coimbatore, INDIA
Email:amshakalacse@yahoo.com

[2]DR.R.NEDUNCHEZHIAN
Department of Information Technology
Sri Ramakrishna Engineering College, Coimbatore, INDIA
Email:rajuchezhian@gmail.com

Abstract: Integration of data from multiple sources has gained importance as data and the data providers explode at a faster rate. Schema matching is considered an important step in integrating data from multiple sources. Most of the available techniques for automated schema matching require interpretation of attribute names and data values. Such techniques fail when the data sources have incomprehensible attribute names and data values. An alternative schema matching technique, which uses statistics from the schema instances and does not require value interpretations, is proposed in this paper. In this work, functional dependency (FD) relationships between attributes of two schemas are represented in the form of a directed dependency graph. A primitive directed graph matching algorithm is used to find the matching between the two dependency graphs and therefore to find the corresponding attributes of the two schemas. The experimental results show that the proposed approach increases the accuracy of matching as it uses fine grained functional dependency relationships between attributes to compare two schemas.

Key words: Data Integration, Schema Matching, Information Theory, Graph matching, Functional Dependency.

1. Introduction

Today's businesses demand an integrated view of data from different data sources. The data sources are heterogeneous in nature and removing such heterogeneity is important for providing an integrated view of data. Schema level heterogeneity could be resolved using an appropriate schema matching technique. When database schemas for the same domain are developed by independent parties, they will almost always be quite different from each other. These differences are referred to as semantic heterogeneity [1].

Schema matching is a process which takes two schemas as inputs and produces a mapping between elements of the two schemas as the output. It plays a very important role in extensive database applications like heterogeneous database integration [2][3][4][5], data warehousing, electronic commerce, semantic web[1] and P2P data management systems[6]. Schema matching is a challenging task for several reasons.

- First and foremost challenge is that, the same real world entity could have different representations and hence the semantic relationships between the schema elements of the independently developed data sources are unknown.

- Attribute names and data values of different data sources may not have lexical similarities or may not be described using same description language.

There are many occasions when data sources have incomprehensible attribute names and data values (i.e., when the meaning of the attribute or data can not be understood from the attribute name or values respectively). For example consider the two relational tables shown in table 1. The attribute names and the values are encoded using different description languages and the semantics of the attributes and their values are not comprehensible.

NM	CT	CC	PIN
Mike	Nyc	01	0172
Rike	Mh	44	0797
Joe	Edi	01	EH10
Jim	Mh	44	W185

(a)

Nam	Cit	C_Cd	Zip
Ben	Edin	02	123
Jan	Las	02	112
Sean	Unt	04	345

(b)

Table 1: Tables with incomprehensible attributes names and values

It is difficult to identify matching attributes in the above mentioned two tables, using lexical similarities of attributes and their values. Schema matching techniques that do not require data interpretations will be the right solution to be adopted in such cases. The work described in this paper is one such schema matching technique that uses statistical characteristics of the data values identify corresponding schema elements and does not require attribute name or data value interpretations.

1.1 Problem Definition

Schema matching can be defined as the problem of computing relations between pairs of attributes belonging to different relational schemas.

Let $R(a_1,a_2,a_3,\ldots.a_n)$ and $S(b_1,b_2,b_3,\ldots.b_n)$ be two schemas with n attributes each, such that a_i , b_i $(1 \leq i \leq n)$ are their respective i^{th} attribute. Schema Mapping $M_{R \rightarrow S}$ between two schema R and S is a set of pairs $<a_i,b_j>$ which indicates that i^{th} attribute of R matches with j^{th} attribute of S for some $1 \leq i , j \leq n$.

In general, schema matching techniques exploit either schema information or instance-level information and they are broadly classified as follows. [7]

Schema-level matchers only consider schema information, not instance data. The available information includes the usual properties of schema elements, such as name, description, data type, relationship types (part-of, is-a, etc.), constraints, and schema structure [8][9].

Instance-level matchers use instance-level data to gather important insight into the contents and meaning of the schema elements. These types of matchers are typically used along with schema level matches in order to boost the confidence in match results, when the information available at the schema level is insufficient. Matchers at this level use linguistic and constraint based characterization of instances [10].

Hybrid matchers combine several matching approaches. Most of these techniques employ additional information from dictionaries, thesauri, and user-provided match or mismatch information. Hence such techniques help determine match candidates based on multiple criteria or information sources [8][11][12][13][14].

Kang and Naughton [15] introduced a new criterion called data interpretation to classify schema matching techniques as interpreted and un-interpreted schema matching.

Interpreted schema matching: Matching techniques that depend on data interpretation are called as interpreted matching.

Un-interpreted schema matching: Matching techniques that does not depend on data interpretation are called as un-interpreted matching.

The advantage of un-interpreted schema matching techniques is that they do not require value interpretations and the data values do not need to have lexical similarities. Even if different encoding schemes are used between two data source, the statistical characteristics of data can be utilized to perform schema matching [15][16][17]. The schema matching technique that is proposed in this paper uses instance-level information to determine the correlations among the attributes in each table and it is an un-interpreted schema matching technique, because it does not require value interpretations of data.

The following are some of the concepts that are used in the schema matching techniques discussed in this paper.

Attribute Entropy [18] - Let X be an attribute in a table and let the probability distribution of X be p(x). Attribute entropy of X is defined as

$$H(X) = -\sum_{x \in X} p(x).\log_2 p(x) \quad (1)$$

Entropy measures the amount of information in an attribute. The entropy is a nonnegative function, i.e., $H(X) \geq 0$. It may be interpreted as a measure of the information content of, or the uncertainty about, the attribute X. Entropy depends on the probabilities, and does not depend on the actual values taken by attribute X.

Mutual information [18] - Let X and Y be two attributes in a table. Consider some joint probability distribution p(x, y) and marginal probability distributions p(x) and p(y) over two attributes X and Y respectively. The mutual information I(X : Y) between X and Y is defined as

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \frac{\log_2 (p(x,y))}{p(x).p(y)} \quad (2)$$

The measure of deviation of the joint distribution from the independence distribution is in fact the mutual information I(X; Y) between the two attributes X and Y. It is non -negative and symmetric, i.e., $I(X; Y) \geq 0$ and $I(X; Y) = I(Y; X)$.

Dependency graph[15]- For the given schema instance $S(a_1,a_2,a_3 \ldots a_n)$ with n attributes, the dependency graph can be represented as a graph with n nodes. In Kang and Naughtan's approach, dependency graph is a weighted graph. The weight of the edge connecting two nodes is the mutual information between two attributes. In the proposed approach, dependency graph is a directed graph with the directed edges indicating functional dependency relationship between attributes.

Probability mass function (pmf) of a distribution is defined as a function that gives the probability that a discrete random variable is exactly equal to some value. For each attribute, the probability mass function is estimated based on the frequency of occurrence counts taken over the available database records, e.g., $P(X = x_1) = N(x_1) / N$, where $N(x_1)$ is the number of times $X = x_1$ occurs and N is the total number of database records (assuming for each record there is a measured value for attribute X)[16].

To provide a better insight on two of the existing un-interpreted schema matching techniques namely Kang & Naughton's and Anuj Jaiswal et. al's schema matching methods, a brief description is included below.

Kang-Naughton's method, pairs attributes in two schemas based on the closeness in attribute entropies and mutual information between attributes. This method is a kind of un-interpreted matching technique which uses attribute entropy and mutual information to represent schema as weighted dependency graph. Each attribute is a specific node in the graph and the weight on the edges connecting two nodes is the mutual information between the attribute pairs. Several techniques were followed to minimize the Entropy-only Euclidean distance metric (defined below), between attribute pairs in the two schemas [15]. Kang and Naughton also performed weighted graph matching by considering, mutual information between the schema elements as weights between the adjacent nodes in the dependency graphs. They used the Euclidean Distance Metric to measure the distance between the two graphs. There are scenarios where Kang and Naughton's schema matching methods are not effective because

- The entropy difference may not be large enough to make confident matching decisions and cardinality (number of distinct values) of the matching attributes are important to get close entropy values.

- One cannot always use value cardinalities to aid matching because it is possible that for two attributes to have same value cardinalities, but their ground truth may not match.

To make progress in such difficult scenarios, Anuj Jaiswal et. al, proposed an un-interpreted schema matching technique in [16], that utilizes value-mapping dimension to enhance schema matching. They believe that probability mass function (pmf) is in general much more distinctive than the attribute entropy or mutual information.

There are also cases were this technique has limitations. First, even if two attributes to be matched do in principle correspond to a matching pair, their pmfs will differ due to some hidden factor. For example, when the value of an attribute is conditioned on another attribute's value and if the condition attribute is unavailable in the database, the pmfs of the matching attributes may greatly differ.

Another limitation of this technique is that, even if two attributes have similar pmfs on their value distributions, the ground truth may not match. The proposed technique uses fine grained functional dependency relationship existing between attributes to construct the dependency graphs. Existence of functional dependencies does not depend either on the value cardinalities or the number of tuples taken in the sample. Even when two attributes have similar pmfs, the functional dependencies that the attributes participate differs. Using functional dependency relationship helps unambiguous matching of attributes. This work focuses on finding a one-to-one mapping where each attribute of a table is mapped to one and only one attribute of another table and this method resolves the ambiguity in making schema matching decision. The primary contributions of this paper are as follows:

- Representing functional dependencies existing between attributes as a directed graph.
- A novel schema matching technique that uses functional dependencies between attributes to identify the structural similarity between the two is proposed.
- Schema matching is done by testing directed dependency graphs for isomorphism
- It is shown through experimental results that the proposed approach produces more accurate schema matches than the existing approaches.

The remaining part of this paper is organized as follows. Works related to schema matching are discussed in section 2. Section 3 describes the proposed schema matching technique. The experimental results are shown in section 4. Section 5 discusses the future work and concludes the paper.

2. Related Work

There is a lot of previous work on schema matching developed in the context of schema translation and integration, knowledge representation, machine learning, and information retrieval [7]. For good surveys and classifications of schema matching methods, see [1][7][29]. As the proposed schema matching technique does not require data interpretations, it can combine with existing schema matching techniques and complement the results produced. Two of the recently proposed works namely Kang and Naughton's inter-attribute dependency based schema matching method [15] [17] and Miller et. al's pmf[16] based schema matching method are also categorized under un-interpreted schema matching technique. But both these schema matching methods have lot of limitations. The entropy difference may not be large enough to make confident matching decisions and the cardinality of the matching attributes are important to get close entropy values and hence Kang-Naughton's approach works well only on data sets with highly varying attribute entropies. Functional dependency is considered as a finer metric than pmf to distinguish attributes in the schemas. Miller et.al 's approach also fails on data sets with close entropy values , because close entropy values are results of close probability distributions of attribute values.

Another schema matching technique called Similarity Flooding is proposed in [11], which represents a schema in a directed labeled graph format and performs matching based on the structural similarity of the two graph representations. The technique starts from string based comparison of the vertices' names to obtain an initial mapping. Depending on the matching goal, a subset of the mapping is chosen using filters. After the algorithm is run, a human is expected to check and if necessary adjust the results. Even though similarity flooding technique uses graph matching algorithm for schema matching, it differs from the proposed work in several ways. It is a type of interpreted schema matching technique that requires interpretations of schema elements. It is a semi-automatic approach unlike the proposed one which does not require human intervention. It is also a kind of hybrid matching technique, since a subset of matching algorithms is used to refine the match results. There are other hybrid schema matching techniques like [8][12][13][14] . Cupid[14] is a hybrid matcher based on both element- and structure-level matching . It is intended to be generic across data models and has been applied to XML and relational examples. COMA[12] schema matching system is developed as a platform to combine multiple matchers in a flexible way and provides a large spectrum of individual matchers, in particular a novel approach aiming at reusing results from previous match operations, and several mechanisms to combine the results of

matcher executions. The main issue with hybrid matchers is how to select the most suitable match algorithms to execute for a given domain. COMA is a framework to comprehensively evaluate the effectiveness of different matchers and their combinations for real-world schemas. Another hybrid matching approach that evaluates performance of several match algorithms is described in [13] which have a matching engine that makes use of a decision tree to combine most appropriate match algorithms.

Clio [30], introduced by Miller et al. performs schema mapping in an interactive fashion using user feedback. Schema matching techniques are roughly classified in [1] into two groups namely Rule-based and Learning-based approaches. Cupid [14] and similarity flooding [11] are two well known rule based hybrid matching techniques. These techniques are relatively inexpensive and do not require training as in learning-based techniques. The main drawback of rule-based techniques is that they cannot exploit data instances effectively, even though the instances can encode a wealth of information. Learning – based solutions have considered a variety of learning techniques and exploited both schema and data information [3] [4] [8] [9]. For example, the SemInt system [9] uses a neural network learning approach and matches schema elements based on attribute specifications (e.g, data types, scale, the existence of constraints) and statistics of data content. The iMAP[8] exploits a variety of domain knowledge, including past complex matches, domain integrity constraints, and overlap data. Finally, iMAP introduces a novel feature that generates explanation of predicted matches, to provide insights into the matching process and suggest actions to converge on correct matches quickly.

Besides research on schema matching, the problem of data matching is also considered crucial for data integration applications. In slightly ironic fashion, the same problem has multiple names across research communities. In the database community, the problem is described as merge-purge [31], record linkage [32], duplicate detection[33][34], and value mapping[16][35]. All the data matching techniques except Miller et al's [16] require data interpretation for duplicate record detection and another variation of this technique is that it uses value mappings to match schema attributes. There is another technique described in [10] that uses duplicates to perform schema matching. In [34] , Elmagarmid et al. have presented a detailed survey on various record matching techniques. Although substantial amount of research has been done in the area of schema matching, still there are issues like handling uncertainties in schema matching and performing matching at a larger scale that are yet to be addressed.

3. Proposed Work

In this section functional dependency based schema matching algorithm is described. The two schemas that are to be matched are passed as input and the matched pairs of attributes are returned as output. The proposed approach only considers the functional dependency relationship between the attributes of the two schemas to be matched and does not require understanding of attribute name or data values. The algorithm works in two steps. In the first step, the functional dependencies existing between attribute of the given tables are extracted and represented as directed functional dependency graphs. In the second step, a directed graph matching algorithm is applied on the dependency graphs to identify the matching attributes of the two schemas.

3.1 Preliminaries

To construct a functional dependency graph, functional dependencies between attributes of the schemas are extracted. Two information theoretic measures namely mutual information and attribute entropy defined in section 1, are used to extract the functional dependencies existing between attribute pairs. The intuition behind using mutual information to extract functional dependency is that it captures complex correlations between attributes by a single number, which simplifies the extraction process. There are several other methods like TANE[19], FUN[20], FD-MINE[21], etc that are used to extract functional dependencies in relational tables. But all these methods use set theory operations like subset computation and comparisons, which are computationally time consuming. Functional dependency (FD) between two attributes is defined as follows.

Functional dependency- A functional dependency $X \rightarrow Y$ holds over relation R if, for every allowable instance r of R, $t_1 \in r$, $t_2 \in r$, $\prod_X (t1) = \prod_X (t2)$ implies $\prod_Y (t_1) = \prod_Y (t_2)$. i.e., given two tuples in r,

if the X values agree, then the Y values must also agree(X and Y are *sets* of attributes)[22].

The algorithm for matching two schemas is shown below. The algorithm takes two schemas R and S as input and returns a match matrix M. The elements of M, m_{ij}=1 when i^{th} attribute of R matches j^{th} attribute of S or m_{ij}= 0 otherwise.

3.2. Functional Dependency Extraction

Functional dependency captures the dependency between attributes. A functional dependency is said to occur between two attributes when one attribute in a relation uniquely determines another attribute. This can be written as X → Y which would be the same as stating "Y is functionally dependent upon X.". If X → Y, the partition of the database by X and Y is the same as the one produced by X alone. In terms of information-theoretic measures, X→ Y holds if and only if

$$H(X, Y) = H(X) \quad (3)$$

Where H(X,Y) is the joint entropy of attributes X and Y and H(X) is the attribute entropy of the attribute X[18][23][24]. By computing attribute entropies of all the attributes and joint entropy between all attribute pairs in the given table, all those left and right reduced functional dependencies (FDs with single left and right hand side attributes) that are true can be determined. This small set of functional dependencies is sufficient to distinguish attributes one another.

A	B	C	D
a1	b2	C1	d1
a3	b4	c2	d2
a1	b1	c1	d2
a4	b3	c2	d3

(a)

W	X	Y	Z
w2	x1	y1	z2
w4	x2	y3	z3
w3	x3	y3	z1
w1	x2	y1	z2

(b)

Table 2: (a) Relational Table R (b)Relational Table S

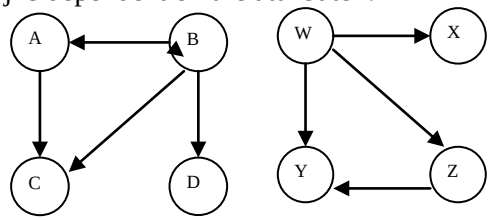
By using equation 6, we can infer the FDs that hold for R and S shown in Table 2. Table 3 shows the FDs inferred from tables R and S.

Table	Inferred FDs
R	A→C, B→A, B→C, B→D
S	W→X, W→Y, W→Z, Z→Y

Table 3: FDs inferred from tables R and S

3.3. Modeling Dependency Relation

The FDs extracted from the tables are used to construct directed functional dependency graphs as shown in Figures 2(a) and 2(b). It is seen that mutual information between A and B is equal to entropy of A, and hence A is fully functional dependent on B and a directed edge is included from B to A. Similarly FDs existing between other pairs of nodes are determined. The dependency graph is represented as directed graph instead of un-directed graph in order to reduce the search space. The functional dependency relationship between any two attributes i and j is represented as a directed edge from i to j, where j is dependent on the attribute i.



(a) **(b)**

Figure 2: Dependency graph for table R, S

3.4 Testing Isomorphism for Directed Graphs

The dependency graphs generated in the previous step are taken as input for directed graph matching algorithm. The graph matching algorithm tests for isomorphism between the two graphs and produces a mapping between corresponding nodes in the two graphs. There are several other graph matching algorithms like [11] [26], but the algorithm proposed in [25] is chosen because it is very primitive and sufficient to get good matching results.

3.4.1 Graph Isomorphism

Given a pair of graphs, G_1 and G_2, isomorphism is a one-to-one mapping φ from the vertices of G_1 onto the vertices of G_2 such that φ preserves adjacency and non-adjacency of the vertices. In terms of the adjacency matrix, two graphs G_1 and G_2 are isomorphic if a permutation of the rows and corresponding columns of adjacency matrix A_1 will produce the adjacency matrix A_2 [25].

For any graph G_1 to be isomorphic to G_2, G_1 must exhibit the same degree sequences as G_2. This is a necessary but not sufficient condition for isomorphism. A degree sequence of graph is merely a listing of the degrees. In-degree and out-degree sequences can similarly be defined. In terms of the adjacency matrix, the degree sequence can be generated by summing the rows and columns corresponding to each vertex. For example, the out-degree sequence, the in-degree sequence, and the degree sequence for Graphs G_1 (Figure 2(a)) are (1,3,0,0), (1,0,2,1), and (2,3,2,1), respectively and that for graph G_2 (Figure 2(b)) are (3,0,0,1),(0,1,2,1), and (3,1,2,2), respectively.

Mapping nodes based on degree sequence leaves some uncertainty about the mappings for vertices A, C and Y, Z. To resolve ambiguities in mapping, we can form characteristic matrix for each graph and map vertices which exhibit identical rows of characteristic matrix. This approach enables a finer mapping between vertices compared to using degree sequences. The characteristic matrix is formed by composing the corresponding rows and columns of row characteristic matrix and column characteristic matrix respectively [7] that are defined as follows.

Row Characteristic Matrix: A *row characteristic matrix XR* is an N × N -1 matrix such that the each element xr_{vm} is the number of vertices which are at a shortest distance m away from v.

Column Characteristic Matrix: A *column characteristic matrix XC* is an N × N-1 matrix such that each element xc_{vm} *is* the number of vertices from which v, is at a shortest distance m.

Characteristic Matrix : A *characteristic matrix C* is an N × N-1 matrix that is formed by composing the corresponding rows and columns of *XR and XC*. Figure 3 shows the adjacency matrices A_1, A_2 of the graphs G_1 and G_2. For any two nodes i, j if there is a directed edge from i to j, $A_{ij} = 1$ otherwise $A_{ij} = \infty$ and $A_{ii} = 0$ (diagonal elements);

A1 A2

$$\begin{pmatrix} 0 & \infty & 1 & \infty \\ 1 & 0 & 1 & 1 \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 1 & 1 & 1 \\ \infty & 0 & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & 1 & 0 \end{pmatrix}$$

Figure 3: Adjacency matrix of Graphs G1 and G2

The row characteristic matrix and the column characteristic matrix for the two graphs G_1 and G_2 are constructed from the adjacency matrices A1 and A2 respectively and are shown in Figures 4 (a) and 4(b) respectively.

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Figure 4(a): Row characteristic matrices of G1 and G2

$$\begin{pmatrix} 3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Figure 4(b):Column characteristic matrices of G1 and G2

The characteristic matrices C_1 and C_2 formed by composing the corresponding rows and columns of row characteristic matrix and column characteristic matrix of the respective graphs G_1 and G_2 are shown in Figure 5.

C_1 C_2

$$\begin{pmatrix} 11 & 0 & 0 \\ 30 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 30 & 0 & 0 \\ 1 & 0 & 0 \\ 2 & 0 & 0 \\ 11 & 0 & 0 \end{pmatrix}$$

Figure 5: Characteristics matrix of Graphs G1 and G2

In a functional dependency graph, a direct edge is included for every functional dependency that holds true between any pair of nodes. The shortest distance

between any two nodes is always 1, if there is an edge (Functional dependency) connecting the two nodes. From Figure 5, it is seen that the graph matching problem is reduced to matching of element of the first column of the characteristic matrices. Section 2.5 explains the schema matching technique using directed graph matching algorithm.

3.5 Directed Graph matching Algorithm for Schema Matching:

The primitive graph matching algorithm helps in matching two graphs by pairing nodes that have identical first column elements in characteristic matrices C_1 and C_2. According to this algorithm, the following node pairs (A, Z), (B, W), (C, Y), (D, X) are identified as matching pairs. As the nodes of the graphs G_1 and G_2 represent attributes of the tables R and S, the matching node pairs is equivalent to the matching attribute pairs. This matching is shown in the schema match matrix M in Figure 6.

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} W & X & Y & Z \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Figure 6: Schema match Matrix M

Match matrix M is a square matrix with $m_{ij} = 1$ when i^{th} attribute of schema R matches with j^{th} attribute of schema S and $m_{ij}=0$ otherwise.

The algorithm for schema matching using directed graph matching is given below. The algorithm takes as input the two schemas R and S for matching and returns the match matrix M as output.

The time complexity of forming the characteristic matrices is $O(N^2)$ where N is the number of attributes in the two schemas that are to be matched. Graph matching by comparing the first column elements of characteristics matrices requires $O(N^2)$ time. The time complexity of graph isomorphism problem is reduced from O(N.N!) to $O(N^2)$, since the proposed work requires only comparison of first column elements of the characteristic matrices.

Algorithm 1 : FD schema matching approach
Input: Schemas R and S
Output: Schema match matrix M
Begin
numOfNodesR ← Number of attributes in R
numOfNodesS ← Number of attributes in S
adjMatrix1 ← etAdjacencyMatrixOfFdGraph(R);
adjMatrix2 ←getAdjacencyMatrixOfFdGraph(S);
charMatrix1←getCharacteristicsMatrix(adjMatrix1);
charMatrix2←getCharacteristicsMatrix(adjMatrix2);
rNodeIndex ← 0
while (rNodeIndex < numOfNodesR)
begin
sNodeIndex ← 0
while sNodeIndex < numOfNodesS
begin
if (charMatrix1[rNodeIndex][0] = =
charMatrix2[sNodeIndex][0]**)**
M[rNodeIndex][sNodeIndex] = 1
end if
sNodeIndex+
end while;
rNodeIndex ++
end while;
return M;
End;

4. Experimental Results

In this section, the proposed schema matching technique is compared with three of the existing approaches namely Kang-Naughton's entropy based schema matching(labeled as KNE in the graphs), Kang-Naughton's mutual information based schema matching(labeled as KNMI in the graphs) and Anuj Jaiswal et.al's first order dissimilarity(labeled as FOD in the graphs) metric based schema matching approach. Our own implementation of the schema matching algorithms is used for the experiment. As Kang-Naughton in [15] [17] used naïve exhaustive search algorithm to find the best schema match, the same is followed in order to get accurate matching results. Anuj Jaiswal et.al's pmf based schema matching algorithm [16] is implemented with embedded value mappings. The proposed functional dependency based schema matching algorithm (labeled as FD in the graphs) is implemented by constructing a directed functional

dependency graph and applying directed graph matching algorithm.

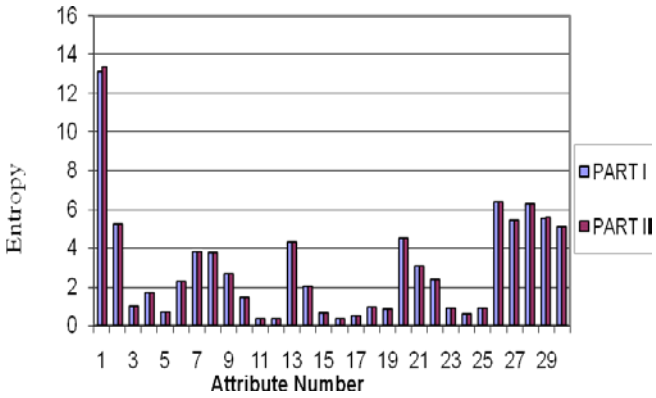
Section 3.1 describes the experimental setup in which the algorithms are implemented and tested. In section 3.2 the data sets used in the experiments are described. Section 3.3 discusses the evaluation metrics used for comparing the performance of the algorithms. Section 3.4 presents the experimental results.

4.1 Experimental setup

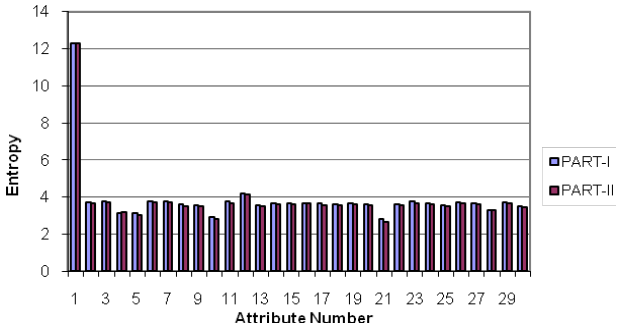
The algorithms are implemented in java and tested in Pentium IV, 1.60 GHz processor and 512 MB RAM on a Windows XP Platform.

4.2 Data Set

Real-world data sets from two different domains are used to test the schema matching algorithms experimentally. One data set is from the medical domain containing data used for diagnosing Meningitis disease and is donated by Dr. Shusaku Tsumoto (Department of Medical Informatics, Shimane Medical University) [27]. It has 38 attributes and 140 instances of test results. In order to increase the number of tuples in the data set to 20K tuples, the 140 instances are randomly replicated and added to the data set. The second data set is US summary data set retrieved from US Census Bureau [12]. This data set contains about 250 attributes and around 15K tuples. For the experiments, 5 to 30 attributes and 10K tuples are selected randomly from the dataset to form two sub tables on each dataset. The two subtables are considered as two different schemas to match, so that we know the correct matching between the schema attributes. The schema matching techniques are repeatedly executed for several iterations by considering the sub tables as two different data tables and the average results are considered for performance analysis. The matching attributes in the sub tables have close entropies in both the data sets but the attribute entropies within each sub table highly varies in the first dataset whereas it does not vary highly in the second dataset. Figure 7(a) shows the attribute entropy distribution of the two partitions of the medical data. It is seen that the attribute entropies of the matching pairs are very close to each other and each attribute in the sub tables have highly varying entropies. Figure 7(b) shows the attribute entropy distribution of the two partitions of the census data. It is seen from the bar chart that almost all the attributes in both the sub tables have close attribute entropies, approximately equal to 3.6.



(a)



(b)

Figure 7: Entropy Distribution: (a) Medical Data, (b)Census Summary data

Datasets having large entropy differences in the attribute entropies among the matching pairs in the two sub tables are not considered for the experiments because, such datasets will not favor Kang-Naughton's Entropy based approach.

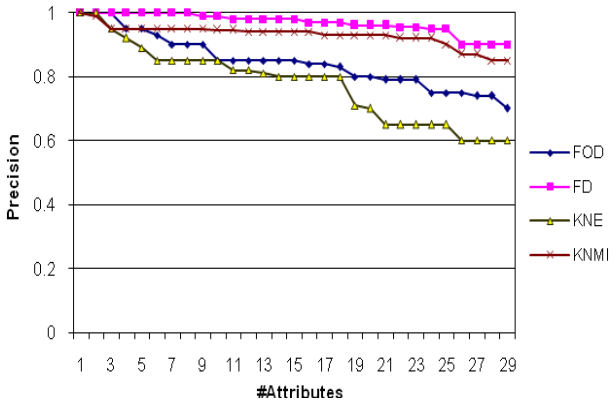
4.3 Evaluation metric

Accuracy of the results returned by the schema matching algorithms are measured using two metrics namely precision and recall. Precision is the ratio of the number of correct matches produced by

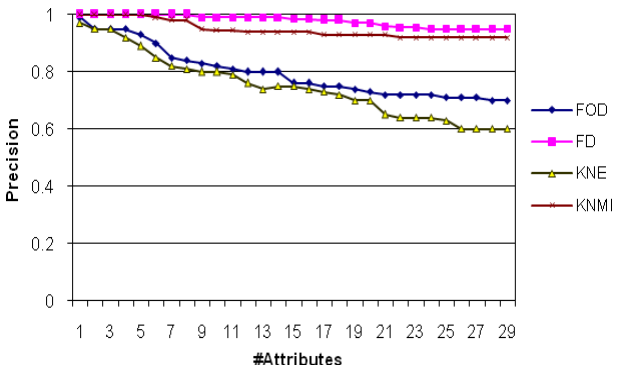
the algorithm to the total number of matches produced by the algorithm. Recall is the ratio of the number of correct matches produced by the algorithm to the total number of correct matches. The number of correct matches is equal to the number of attributes in the two schemas taken for matching, since we treat two sub tables of the same data set as two different schemas. Computational time is another important metric to compare the efficiency of the schema matching algorithms. Computational time varies as the number of tuples and attributes are varied in the data sample taken for experiments. The performance analysis between the schema matching techniques is explained in section 3.4.

4.4 Performance analysis

Figure 8(a) shows the precision of the results obtained by running all the four algorithms considered in this work using the medical data set. The precision of the proposed algorithm (FD in graph) and Kang-Naughton's KNMI algorithm remains stable even if the number of attributes are gradually increased, whereas the precision of the pmf (FOD in graph) based approach and Kang-Naughton's entropy based algorithm(KNE) decreases with the increase in the number of attributes.



(a)Medical Data set



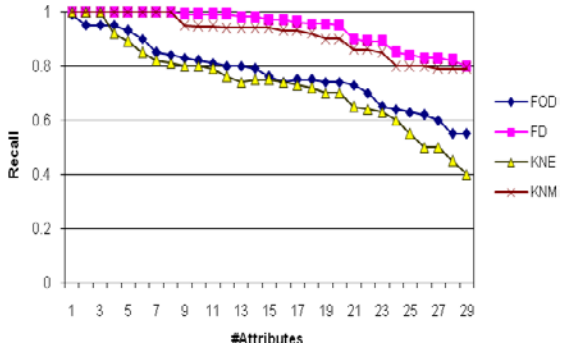
(b) Census Data set

Figure 8: Precision Vs Number of attributes

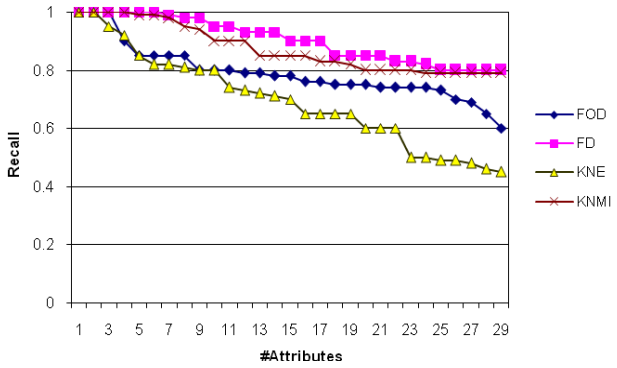
Since the medical data set has attributes with varying entropies among them within the tables, the precision of the matching results produced by FOD based approach is 86% on average and that of the KNE approach is 82% on average. The precision of results produced by FD approach does not fall below 93% and produces results of 95% on average. Kang-Naughton's KNMI approach produced results with a precision of 92% on average. Figure 8(b) shows the precision of the results obtained by running all the four algorithms considered in this work using the census summary data set. The precision of the proposed algorithm (FD in graph) is not less than 90% and that of Kang-Naughton's KNMI algorithm is 87% since Euclidean distance metric based exhaustive search algorithm is used. The average precision of the results produced by FOD and KNE approaches for the census summary dataset fall below 80% and 70% respectively. Most of the attributes in the census summary dataset has same entropy (approximately 3.6) and also the pmf between various attributes have very small Euclidean distance differences between each other and hence the FOD and KNE approach could not produce accurate matching results. The proposed FD approach and Kang-Naughton's KNMI approach produce results with higher precision because these two approaches use inter attribute relationships to differentiate attributes from one another.

Recall is another performance metric that is used to analyze the completeness of the algorithms. It is the ratio of the number of correct matches returned by the algorithm to the total number of correct

matches. Recall of the results obtained by running all the four algorithms on the medical dataset is shown in Figure 9(a). The results returned by FD based approach has recall little lesser than precision.



(a) Medical Data set



(b) Census Data Set

Figure 9: Recall Vs Number of attributes

Average recall produced by KNMI and FD approaches is 90% and 92% respectively. FOD and KNE approaches produce results with average recall of 82% and 70% respectively. Figure 9(b) shows the recall of the match results retuned by the four algorithms for the census summary data set. Recall of the results produced by KNMI and FD approaches is approximately 85% on average and they remain stable even for wider tables which has more number of attributes. It is seen from the graph that the recall of the KNE and FOD approaches decreases as the number of attributes increase from 2 to 30. The dependency graph may look different for the two sampled tables and may not produce matching results when there is no match between the nodes of the dependency graphs. Recall of FOD approach is below 60% when 30 attributes are considered. On average recall of FOD approach is 70% and that of KNE approach is 65%.

Low precision implies that false correspondences have to be manually deleted by the user, while low recall indicates that missing correspondences have to be manually added [44]. For instance, for the census summary dataset FOD based approach produced match results with 70% recall and 80% precision when 15 attributes are considered. 70% recall means that only 10 matches out of the 15 true matches is being detected by the algorithm and 80% precision means that only 8 out of 10 matches detected by the algorithm is correct. The algorithm has missed 5 true matches and has produced 2 false matches.

The proposed algorithm is also analyzed in terms of computation time. Figure 10 shows the computational time taken by the four algorithms discussed in this paper. KNMI uses branch and bound approach which would obviously be the best in terms of the accuracy but it could be too slow for large problems. The computational time is measured by running the algorithms on the summary data set for several iterations and averaging the results. The data set has 15K tuples. The execution time of the proposed approach is slightly higher compared to FOD approach, because it has to compute inter-attribute dependency (Joint entropy) between all pairs of the two schemas.

The KNE approach is a simple technique which detects matching schema elements by comparing their entropy value. Attribute pairs with minimal difference between their respective entropy is marked as matching pairs and this algorithm takes very less time to produce results. The computation time taken by KNMI approach highly varies with that of FD, FOD and KNE approaches. The execution time of the algorithms is measured by varying the number of attributes from 2 to 12. The time taken by FD approach is lesser than the KNMI approach, because KNMI algorithm requires determining permutation matrices, which is computationally intensive.

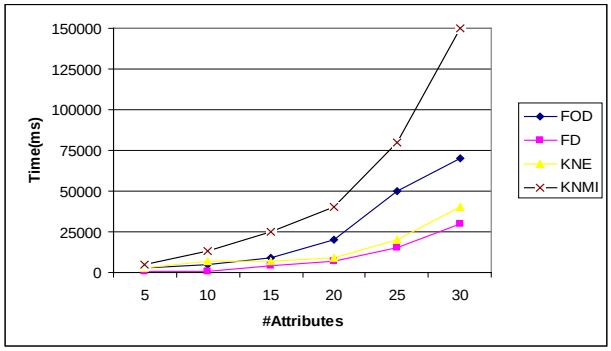


Figure 10: Computational time Vs #attributes

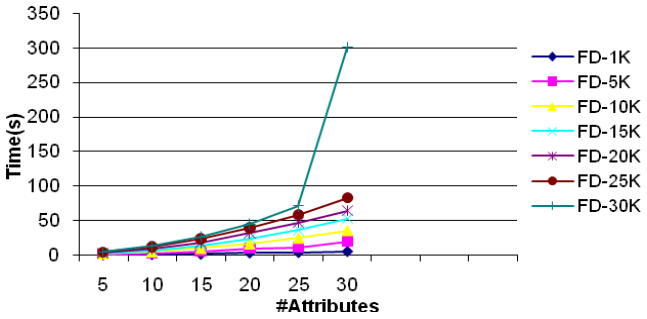


Figure 11: Data Sampling Effects

Figure 11 illustrates the effect of row sizes on computational time(in seconds) required by the FD based schema matching algorithm for the census summary data set. This data set originally had only 15K tuples and it is replicated to increase the size to 30 K tuples. The time taken by the FD algorithm for 5K tuples (labeled FD-5K), 10K tuples (labeled FD-10K), 15K tuples (labeled FD-15K), and 20K tuples (labeled FD-20K) are shown. It is seen that the time complexity of the FD based schema matching algorithm increases as the number of tuples in the data table increases. For the table size below 20 K tuples, the FD algorithm takes less than 100 seconds and it takes maximum time of 300 seconds when the table size is 30 K tuples and the number of attributes is 30.

From the experimental results it is shown that the proposed FD based schema matching algorithm produces results with high precision, equivalent to the results produced by KNMI exhaustive search algorithm for any type of data set. The KNE approach based on entropy and the FOD approach based on pmf of the value distributions produces poor results for data sets that have attributes with very close entropies. Algorithms that consider attribute correlations like mutual information and functional dependency relationship give accurate results compared to the algorithms that consider statistics of individual attributes. Kang-Naughton's Mutual information based approach and the proposed FD based approach produces better results because they use inter attribute dependencies to match schema attributes. The proposed FD approach produces accurate results as that of KNMI approach with much lesser computation time.

5. Conclusion

A schema-matching technique is proposed in this paper that works even in the presence of incomprehensible attribute names and data values. Functional dependencies between attributes in the tables to be matched are extracted using information theoretic measures and a directed dependency graph is constructed. In the next stage, matching node pairs across the dependency graphs are identified by running a graph-matching algorithm. It is shown that, although entropy based schema matching is effective, further improvement is possible by exploiting inter attribute correlations like mutual information or functional dependency. In this work, four algorithms for the schema matching problem are investigated and it is proved experimentally that the algorithms using relationships existing between attributes produce better results compared to the ones using individual attribute's value distribution. The proposed approach uses fine grained functional dependency relationships and produces accurate results with much lower computation time compared to the methods using entropy, mutual information or pmf of attributes. There exist several open issues to be addressed in the future. Some of them are, reasoning about imprecise matching results, handling dynamic environments where the data source changes quite often and performing schema matching over very large datasets.

References

[1] A. Doan, Alon Y. Halevy, "Semantic Integration Research in the Database Community: A Brief Survey ",AI Magazine, Vol. 2, 2005, pp. 83-94.

[2] S. Castano, V.D. Antonellis, and S.D.C. di Vimercati, "Global Viewing of Heterogeneous Data Sources," IEEE Transactions on Knowledge and Data Engineering, vol. 13, no. 2, Feb 2001,pp. 277-297.

[3] W.-S. Li and C. Clifton, "Semantic Integration in Heterogeneous Databases Using Neural Networks,", Proceedings of the 20th International Conference of Very Large Data Base ,1994, pp. 1-12.

[4] A. Doan, P. Domingos, and A.Y. Halevy, "Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach," Proc. ACM SIGMOD, 2001, pp. 509-520.

[5] L. Xu and D. Embley, "Using Schema Mapping to Facilitate Data Integration," EROS, 2003.

[6] A.Y. Halevy ,Zachary G. Ives , Jayant Madhavan ,Peter Mork ,Dan Suciu ,Igor Tatarinov, "The Piazza Peer Data Management System", IEEE Transactions on Knowledge & Data Engineering, vol. 16 , 2004, pp. 787-798.

[7] E. Rahm and P.A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," The VLDB Journal, vol. 10, no. 4, 2001, pp 334-350.

[8] R.Dhamankar , Y.Lee , A. Doan , A. Halevy , P.Domingos, "iMAP: Discovering complex semantic matches between database schemas" in Proceedings of ACM SIGMOD International Conference on Management of Data,2004.

[9] W.-S. Li and C. Clifton, "SEMINT: A Tool for Identifying Attribute Correspondences in Heterogeneous Databases Using Neural Networks," Journal of Data and Knowledge Engineering , vol. 33, No.1, Dec. 2000.

[10] A. Bilke, F. Naumann, "Schema Matching using Duplicates", Proceedings 21st International Conference on Data Engineering, ICDE ,2005, pp.69 - 80.

[11] M. Sergey and G.Molina, Hector and Rahm, Erhard , "Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching". In: 18th International Conference on Data Engineering (ICDE 2002), February 26 - March 1.

[12] H.H Do, E.Rahm, "COMA – A System for Flexible Combination of Match Algorithms", VLDB 2002.

[13] F. Duchateau and Z. Bellahsene and R. Coletta, " A Flexible Approach for Planning Schema Matching Algorithms" , Proceedings of the OTM 2008 Confederated International Conferences, 2008.

[14] J. Madhavan, P.A. Bernstein, and E. Rahm, "Generic Schema Matching with Cupid," Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 49-58, 2001.

[15].Jaewoo Kang, Jeffrey F. Naughton, (2008),"Schema Matching using Interattribute Dependencies", IEEE Transactions on Knowledge and Data Engineering, Vol 20,no.10,pp-1383-1407, Oct 2008.

[16].A.Jaiswal, D. J. Miller, P. Mitra, "Un-interpreted Schema Matching with Embedded Value Mapping under opaque Column Names and Data Values", IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 2, Feb. 2010, pp. 291-304.

[17] J. Kang and J. F. Naughton, "On schema matching with opaque column names and data values", SIGMOD, New York, USA, 2003, pp 205–216.

[18].T.M Cover, J.A Thomas, "Elements of Information Theory ",Wiley-Interscience; 99th edition,1991.

[19] Y. Huhtala, J. Karkkainen, P. Porkka, and H. Toivonen, "TANE:Efficient Discovery of Functional and Approximate Dependencies Using Partitions," Proceedings of 14th International Conference of Data Engineering , 1998.

[20] N. Novelli , Cicchetti R ," FUN: an efficient algorithm for mining functional and embedded Dependencies", Proceedings of the International Conference on Database Theory, London, UK, 2001, pp 189–203.

[21] H.Yao, H.J. Hamilton," Mining functional dependencies from data", Data Mining and

Knowledge Discovery ,Springer Volume 16(2), pp: 197-219 ,2008.

[22]Raghu Ramakrishnan, Johannes Gehrke,"Database management systems",McGraw-Hill,2003.

[23].Y.Y Yao, "Information-Theoretic Measures for Knowledge Discovery and Data Mining", Springer, Berlin pp 115-136, 2003.

[24] C.Giannella, and R.Edward, "On Approximation Measures for Functional Dependencies", Information Systems Archive 29(6),2004,pp: 483-507.

[25] D. C Schmidt, L. E Druffel , "A Fast Backtracking Algorithm to Test Directed Graphs for Isomorphism Using Distance Matrices", Journal of the ACM (JACM),Volume 23 Issue 3, July 1976.

[26] L.P Cordella , P.Foggia, C. Sansone, F.Tortorella, M. Vento, "Graph matching: A fast algorithm and its evaluation" , ICPR '98 Proceedings of the 14th International Conference on Pattern Recognition-Volume 2 ,1998.

[27] http://lisp.vse.cz/pkdd99.

[28] US Census Bureau , ftp://ftp2.census.gov/census_2000/datasets.

[29] P. A Bernstein, Jayant Madhavan, Erhald Rahm, "Generic Schema matching:Ten years Later", VLDB September 2011.

[30] L.Yan, Miller, R.; Haas, L.; and Fagin, R. ," Data driven understanding and refinement of Schema mappings", In Proceedings of the ACM SIGMOD, 2001.

[31] M.A. Hernandez and S.J. Stolfo, "The Merge/Purge Problem for Large Databases," Proc. ACM SIGMOD, 1995,pp. 127-138.

[32] I.P. Fellegi and A.B. Sunter, "A Theory for Record Linkage," Journal of American Statistical Association, vol. 64, no. 328, 1969,pp. 1183-1210.

[33] W. Su, J. Wang, F. H. Lochovsky ,"Record matching over query results from multiple web databases", IEEE Transaction on Knowledge and Data Engg,vol 22,April 2010.

[34] A.K.Elmagarmid , P.G. Ipeirotis, and V.S. Verykios, "Duplicate record detection : A Survey",IEEE Transaction on Knowledge and Data Engineering ,vol.19,no.1,pp.1-16,Jan.2007.

[35] J. Kang, D. Lee, and P. Mitra, "Identifying Value Mappings for Data Integration: An Unsupervised Approach," Proc. Conf. Web Information Systems Engineering (WISE), 2005.