

Singapore Management University
Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

2003

Instance Based Attribute Identification in Database Integration

Ee Peng LIM


Singapore Management University, eplim@smu.edu.sg

Cecil CHUA

Roger Hsiang-Li CHIANG

DOI: <https://doi.org/10.1007/s00778-003-0088-y>

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the [Databases and Information Systems Commons](#), and the [Numerical Analysis and Scientific Computing Commons](#)

Citation

LIM, Ee Peng; CHUA, Cecil; and CHIANG, Roger Hsiang-Li. Instance Based Attribute Identification in Database Integration. (2003). *VLDB Journal*. 3, (3), 228-243. Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/18

This Journal Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Instance-based attribute identification in database integration

Cecil Eng H. Chua¹, Roger H. L. Chiang², Ee-Peng Lim³

¹ J. Mack Robinson College of Business, Georgia State University; E-mail: cchua@cis.gsu.edu

² College of Business Administration, University of Cincinnati; E-mail: Roger.Chiang@uc.edu

³ School of Computer Engineering, Nanyang Technological University; e-mail: aseplim@ntu.edu.sg

Edited by L. Raschid. Received: August 30, 2001 / Accepted: August 31, 2002

Published online: July 31, 2003 – © Springer-Verlag 2003

Abstract. Most research on attribute identification in database integration has focused on integrating attributes using schema and summary information derived from the attribute values. No research has attempted to fully explore the use of attribute values to perform attribute identification. We propose an attribute identification method that employs schema and summary instance information as well as properties of attributes derived from their instances. Unlike other attribute identification methods that match only single attributes, our method matches attribute groups for integration. Because our attribute identification method fully explores data instances, it can identify corresponding attributes to be integrated even when schema information is misleading. Three experiments were performed to validate our attribute identification method. In the first experiment, the heuristic rules derived for attribute classification were evaluated on 119 attributes from nine public domain data sets. The second was a controlled experiment validating the robustness of the proposed attribute identification method by introducing erroneous data. The third experiment evaluated the proposed attribute identification method on five data sets extracted from online music stores. The results demonstrated the viability of the proposed method.

Keywords: Attribute identification – Database integration – Measures of association

1 Introduction

The traditional approach to data management constructs a single database for a particular organizational need. However, because of increasing organizational size and complexity, information needs are evolving from the management of a single set of data toward the integration of various sets of organized data [41,63]. In database integration research, schema integration (e.g., attribute identification) occurs prior to instance integration (e.g., entity identification). We challenge this rigid approach and demonstrate that database integration is more appropriately done in an iterative manner. The objectives of this research are to:

- **Demonstrate that schema information alone is sometimes insufficient for database integration:** to correct this problem, we illustrate that instance information can be used to identify the attributes to be integrated.
- **Demonstrate the feasibility of interleaving instance integration with schema integration:** we demonstrate that it should be possible to perform entity identification (an instance integration task) before attribute identification (a schema integration task).
- **Propose a robust attribute identification method:** our method uses instance information to overcome many known problems in attribute identification. Specifically, our method:
 - *Matches attributes with different data types:* the method should be able to identify that a student's letter grade is the same as his or her percentile grade.
 - *Identifies synonyms or homonyms:* the method should be able to identify that gender and sex are equivalent.
 - *Matches attributes with different scales:* the method should be able to match weight attributes with different measurement units such as pound and kilogram.
 - *Matches attributes where one is an abstraction of another:* the method should be able to identify that letter grade is an abstraction of percentile grade.
 - *Matches attributes having erroneous or null values:* although the method uses instances, it should be robust to errors in the instances.
- **Demonstrate the viability of the method:** three experiments were conducted to evaluate the method.

The main contribution of this research is to propose and validate an instance-based attribute identification method to improve database integration when schema information is either inadequate or inappropriate. There are two other contributions. First, we demonstrate that the relationship between schema and instance integration for database integration can be iterative. Second, we propose a robust method that uses both schema and instance information to perform attribute identification. The proposed method is employed when the tuples of corresponding relations can be successfully matched. A three-step process is then used to perform attribute identification as shown in Fig. 1. First, attributes are classified into domain classes and combined to make attribute groups. The

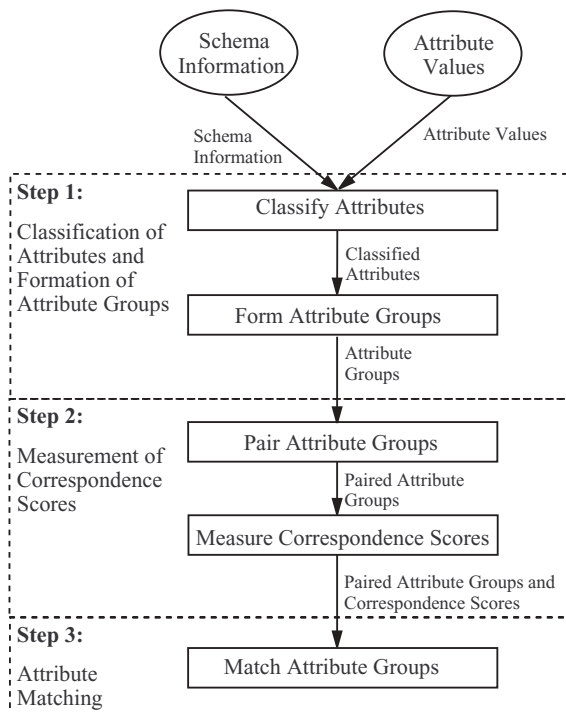


Fig. 1. Attribute identification process

correspondence scores of candidate pairs of attribute groups for integration are then measured. Finally, based on the correspondence scores, the process matches attribute groups for integration. Although our research addresses the attribute identification problem for the case where two relations are being integrated, this can be extended to the N relation case by integrating the N relations in a pairwise fashion. Thus, in a three-relation case with relations A , B , and C , the relations A and B are integrated first. Relation C is then integrated with relation AB .

The paper proceeds as follows. Section 2 reviews related research and motivates our approach. Sections 3 to 5 discuss each step of the process in detail. Section 6 presents three experiments designed to validate the proposed method. Section 7 concludes the research work, discusses the limitations, and describes further research opportunities. Finally, the Appendix presents the heuristic rules developed to perform the attribute identification task in detail.

2 Database integration research

Database integration is often divided into schema integration and instance integration. Schema integration [33] reconciles schema elements (e.g., entities, relations, attributes, relationships) of heterogeneous databases. Instance integration matches tuples and attribute values. Attribute identification is the task of schema integration that deals with identifying matching attributes of heterogeneous databases. Entity identification [43] is the task of instance integration that matches tuples from two relations based on the values of their key attributes. These terms and their definitions are summarized in Table 1.

Table 1. Definitions

	Term	Definition
1	Attribute	One of the fields in a relation
2	Attribute group	A set of attributes from the same relation
3	Attribute identification	The process of matching attributes from corresponding relations in heterogeneous databases
4	Correspondence measurement function	A function that measures the similarity between two attribute groups
5	Domain class	A way of classifying attributes or attribute groups by the properties useful for attribute identification
6	Entity identification	The process of matching tuples from corresponding relations in heterogeneous databases
7	Instance integration	The process of integrating tuples and instances from heterogeneous databases
8	Schema integration	The process of matching and integrating schema elements from different heterogeneous databases

2.1 Attribute identification

Attribute identification is a schema integration task. Existing methodologies for schema integration often assume that information is available for attribute identification [35,57,62]. However, these methodologies do not address how the information can be obtained. There are three main approaches to attribute identification that explore domain knowledge or schema information: (1) the manual approach, (2) the knowledge-based approach, and (3) the data-mining approach.

- **Manual approach:** the manual approach assumes that information required to perform attribute identification (e.g., domain knowledge) is not available [35,57,62]. Humans are required to perform the task of attribute identification but can be guided by a methodological process. This tends to be time consuming, tedious, and error-prone [38].
- **Knowledge-based approach:** the knowledge-based approach attempts to discover matching attributes by their names using a knowledge base, thesaurus, or other “word matching” database [8,9,49,58,61]. The effectiveness of this approach is limited by the amount of useful information accumulated by the knowledge base.
- **Data-mining approach:** this approach uses data-mining algorithms such as neural networks and classification algorithms to compute correspondence scores among attributes [18,39,40,50,54].

Among these approaches, the data-mining approach is the one that can best use the information contained in instances. However, existing data-mining techniques rely primarily on summary instance information (such as mean and standard deviation) for attribute identification. But, as shown in Table 2, the schema and summary instance information do not suggest any corresponding attributes between the tables. For example, $Scr-A$ (i.e., the student’s percentage grade) corresponds to $L-Grd-B$ (i.e., the student’s letter grade), but not to

Table 2. Measuring correspondence using the proportion of explained variance

<i>Relation A</i>						<i>Relation B</i>				
ID-A	Scr-A	Weight-A	Consent-A	Cm-A	Mt-A	ID-B	L-Grd-B	Kgs-B	Expt-B	Pgm-B
1	95	85	Parent	9	9	1	A	39	No	G
2	80	69	Child	3	3	2	B	31	No	N
3	76	61	Both	4	1	3	C	28	Yes	S
4	87	74	None	6	8	4	B	34	No	N
5	73	71	Parent	1	8	5	C	32	No	S
6	91	73	Both	8	9	6	A	33	Yes	G
7	77	87	Parent	5	5	7	C	39	No	N
8	84	77	Both	5	5	8	B	35	Yes	N

(a) Relations with corresponding attributes

<i>Relation A</i>						<i>Relation C</i>				
ID-A	Scr-A	Weight-A	Consent-A	Cm-A	Mt-A	ID-C	Class-C	Stp-C	Fgn-C	Phys-C
1	95	85	Parent	9	9	1	B	35	Yes	S
2	80	69	Child	3	3	2	A	39	No	N
3	76	61	Both	4	1	3	B	31	No	G
4	87	74	None	6	8	4	C	28	No	S
5	73	71	Parent	1	8	5	A	34	No	N
6	91	73	Both	8	9	6	C	32	Yes	N
7	77	87	Parent	5	5	7	B	33	No	N
8	84	77	Both	5	5	8	C	39	Yes	G

(b) Relations without corresponding attributes

Class-C (i.e., the student’s physical education class). However, both L-Grd-B and Class-C have identical schema and summary information (e.g., data types, number of distinct data instances, range of data instances). The only distinction is in their attribute names, which is uninformative. An examination of their instances, however, does reveal the corresponding attributes. For the ID “1,” the Scr-A of 95 maps to the L-Grd-B of “A.” For the ID “2,” the Scr-A of 80 maps to the L-Grd-B “B,” etc., and it becomes clear that a grade-based mapping function could translate one into the other. However, no obvious mapping function would translate Scr-A to Class-C.

Relation A contains student information in a homeroom teacher’s database, Relation B contains student information in the database of the administrative office, while Relation C contains information from the department of physical education. Data in all three relations refer to the same set of students. Attributes in Relations B and C have equivalent data types, data lengths, and instances. The only distinction between these two relations is the order of the instances and their (noninformative) attribute names. In this case, schema and summary instance information is necessary but not sufficient for attribute identification.

However, by matching individual instances, it can be observed that the attributes in Relation A match with attributes in Relation B, but not with Relation C. Scr-A, the student’s final class score, can be mapped to L-Grd-B, the student’s letter grade. Wgt-A is the student’s weight in pounds, while Kgs-B is the weight in Kilograms. Cons-A identifies whether the child, parent, or both consent to having the child undergo experimental educational treatments. Only if both

parent and child consent can the child be used for experiments (Expt-B). Finally, the child’s standardized communications score (Cm-A) and mathematics score (Mt-A) determine whether the child enters the “Gifted,” “Normal,” or “Special Assistance” program Pgm-B. The physical education class number of the child (Class-C), the number of situps the child performs in 1 min (Stp-C), whether the child is a foreigner (Fgn-C), and the physical ability of the child (Phys-C) do not correspond to attributes of Relation A.

2.2 Entity identification

Attribute identification in Table 2 assumed that entity identification could be performed. In many cases, entity identification can be performed with only partial schema information. For example, tuples can be matched by matching known key attributes. In the example, the student’s ID was used universally throughout the school. There are many cases where matching key attributes is sufficient for matching tuples such as:

- *Relations that describe personnel records:* people are often referred to by a standard serial number, e.g., Social_Security_Number and Employee_No.
- *Relations that have well-accepted keys:* examples include relations that describe books (identified by ISBN) or Internet hosts (identified by IP_Number).

Even when matching candidate keys do not exist, alternative ways to match tuples have been suggested such as probabilistic matching of keys and matching of approximate keys [15, 19, 27, 42, 43].

3 Attribute classification and group formation

This section presents the first step of the attribute identification process: the classification of attributes and formation of attribute groups. Heuristic rules have been developed to assign domain classes to attributes (i.e., attribute classification) based on the schema information and the analysis of attributes' values. The classified attributes then form groups for integration.

3.1 Domain class hierarchies

We have identified various properties that are useful for determining both the appropriate summary instance information and the way to analyze instances for attribute identification [31,43,59]. From these properties, we established three *domain hierarchies*, STRING, CODED, and KEY, as shown in Fig. 2. Each domain class within the hierarchies has unique properties.

- **Key domain hierarchy:** the KEY domain hierarchy has key properties. There are two domain classes in this domain hierarchy, CANDIDATE KEY and FOREIGN KEY.
- **String domain hierarchy:** the STRING domain hierarchy has string properties with three different classes, ALPHABETIC, ZERONINE, and MIXED. The ZERONINE domain class is for attributes with only numeric characters (e.g., “0” . . . “9”) in their values. The ALPHABETIC domain class is for attributes with only nonnumeric characters in their values. The MIXED domain class is for attributes with both numeric and nonnumeric characters in their values.
- **Coded domain hierarchy:** the CODED domain hierarchy organizes mathematical/statistical properties (e.g., distinctness, distance, order) with three main domain classes.
 - In the NOMINAL domain class, distinct values are not equivalent. For example, “Parent” and “Child” in the attribute `Cons-A` are different from each other. The NOMINAL domain class has two terminal classes, CATEGORICAL and DICHOTOMOUS. The CATEGORICAL domain class is for attributes with several distinct instances. For example, the attribute `Cons-A` can be assigned the CATEGORICAL domain class. If an attribute's values has only two distinct values, it should be assigned the DICHOTOMOUS domain class. DICHOTOMOUS is separated from CATEGORICAL because these kinds of attributes have unique statistical properties [29] that can be used for attribute identification.
 - The ORDINAL domain class has the distinctness and *order* properties. Values of an attribute with this domain class can be ranked. For example, it is appropriate to say for `L-Grd-B` that an “A” is better than a “B.” However, it is not possible to quantitatively determine how much better an “A” is.
 - The INTERVAL domain class has all the properties of the ORDINAL one. In addition, this domain class conveys the property of *distance*. It has two subclasses, the DATE and NUMBER domain classes. The DATE domain class is used for attributes whose values indicate dates, whereas the NUMBER domain class is established for those attributes whose values are subject to all arithmetic operations (e.g., multiplication, division).

3.2 Attribute classification

The domain classes are assigned to attributes according to information obtained from the (1) database schema such as data type, key information, display format, and data domain, (2) summary information such as mean and standard deviation, and (3) the results of data analysis on attribute values.

An attribute's values may satisfy properties that belong to multiple *domain hierarchies* but should satisfy the properties of at least the STRING domain hierarchy. *Terminal classes* are the only classes assigned to attributes. For example, based on the attribute values of `Salary`, the NUMBER and ZERONINE domain classes can be assigned to `Salary`. To reduce the computational effort, only one domain class per attribute is assigned. Therefore, domain classes are assigned to an attribute in the sequence KEY, CODED, and STRING. This domain class assignment sequence is established for two reasons:

- Key information and character string information are useful for assigning CODED domain classes. For example, if an attribute contains alphanumeric strings, it is unlikely to be a NUMBER.
- Key information is easier to derive than character string information since it is often directly embedded in schema information.

Key domain classes: Assigning attributes with KEY domain classes is performed by reading the data dictionary, deriving it (e.g., using the algorithms proposed by [11,24,45]), or obtaining it manually. Attributes having both the candidate key and foreign key properties are assigned the CANDIDATE KEY domain class since entity identification implies that all candidate keys match. Foreign key properties become superfluous for attribute identification once an attribute is matched as a candidate key.

Coded domain classes: Attributes with a particular data type can have many possible CODED domain classes. For example, an attribute with the Integer data type can be assigned the NUMBER domain class (e.g., `Salary`), the DATE domain class (e.g., `Military_Date`), the ORDINAL domain class (e.g., `Grade` where “1” is an “A,” “2” a “B,” etc.), the CATEGORICAL domain class (e.g., `Consent`, where “1” is “Both,” “2” is “Parent,” etc.), or the DICHOTOMOUS domain class (e.g., `Foreigner` where “1” is “Yes”).

We have established a set of heuristic rules (extensions of work done in [12,55]) to classify attributes into the CODED domain classes. These rules use both schema and instance information to classify attributes. For example, dichotomous attributes can be distinguished either because they have a **boolean** data type, or have two or fewer unique instances (e.g., `Gender`). Some kinds of CATEGORICAL attributes are often coded as words (e.g., `Occupation` or `Religion`). These can be distinguished by comparing their instances to a publicly available electronic dictionary such as WordNet [48]. A full list of rules is presented in Appendix A.1.

String domain classes: the assignment of STRING domain classes is performed based on analysis of the instances' char-

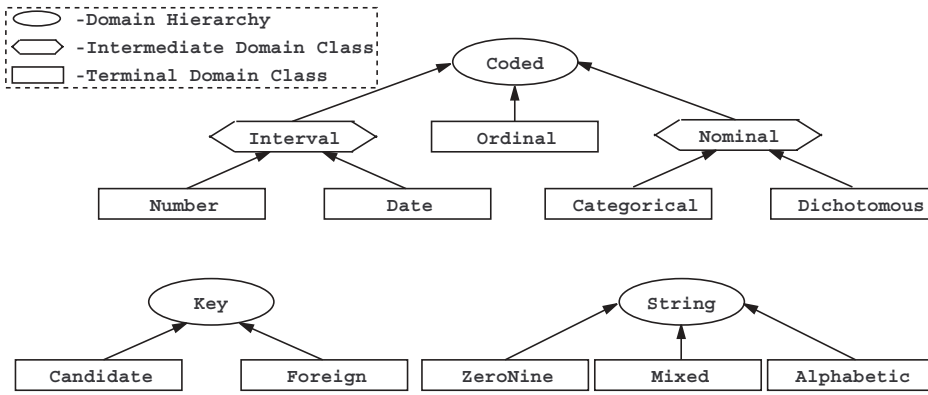


Fig. 2. Domain class hierarchies

acters. For example, instances with only characters ranging from 0 to 9 are assigned the ZERO NINE domain class.

Because these are heuristic rules, it is possible that an inappropriate domain class could be assigned. Any inappropriate domain class assignment can be reviewed and changed manually by the database integration specialist.

3.3 Formation of attribute groups

Unlike other attribute identification methods that match single attributes, our method matches attribute groups. A set of attributes is grouped together if a valid domain class can be assigned to the group. For example, CODED and STRING domain classes can only be assigned to attribute groups if all the attributes in that group have CODED and STRING domain classes. KEY domain classes can be assigned to attribute groups irrespective of the domain classes of the individual attributes. For example, the attribute group {First_Name, Last_Name} may be assigned the CANDIDATE KEY domain class, even though the individual attributes were not assigned a KEY domain class.

The assignment of terminal domain classes from the KEY and STRING domain hierarchies to attribute groups is performed in a similar manner to that with individual attributes. Terminal domain classes of the CODED domain hierarchy are assigned to attribute groups based on the assignment rules presented in Table 3. Assigning appropriate domain classes from the CODED domain hierarchy to attribute groups is an associative process starting from two member attributes. The sequence of combination is immaterial for the resulting domain classes. For example, an attribute group has three attributes – L-Grd-B, Expt-B, and Pgm-B – assigned the ORDINAL, DICHOTOMOUS, and ORDINAL domain classes, respectively. Referring to Table 3, the domain class that results from combining L-Grd-B and Expt-B is CATEGORICAL. {L-Grd-B, Expt-B} combined with Pgm-B is also CATEGORICAL. Were the sequence of combination changed (e.g., if the domain class of {L-Grd-B, Pgm-B} were derived first), the final domain class would still be CATEGORICAL. The justification of these assignment rules presented in Table 3 is discussed in Appendix A.2.

Table 3. Summary of CODED domain class assignments

Grp A	Grp B				
	NUM ^a	DATE	ORD ^b	CAT ^c	DICH ^d
NUM ^a	NUM	DATE	N/A	N/A	NUM
DATE	DATE	DATE	N/A	N/A	DATE
ORD ^b	N/A	N/A	CAT	CAT	CAT
CAT ^c	N/A	N/A	CAT	CAT	CAT
DICH ^d	NUM	DATE	CAT	CAT	DICH

^a NUMBER
^b ORDINAL
^c CATEGORICAL
^d DICHOTOMOUS

4 Measurement of correspondence scores

Once the attribute groups are formed, the attribute identification process performs the following tasks.

1. *Pair the attribute groups.* All attribute groups from Relations A and B that have the same domain hierarchy are paired together. For example, the pair {{Cons-A}, {Expt-B}} is established since {Cons-A} and {Expt-B} have been assigned CATEGORICAL and DICHOTOMOUS, respectively, and both are under the CODED domain class hierarchy.
2. *Measure the correspondence scores.* For each pair of attribute groups, the appropriate measurement function is determined based on the domain classes. The chosen measurement function is then applied to the pair to produce a correspondence score. This task is elaborated in the following three subsections.

4.1 KEY correspondence score measurement

Correspondence scores of attribute group pairs with CANDIDATE KEY and FOREIGN KEY domain classes are determined in the following manner.

1. *Both attribute groups are assigned CANDIDATE KEY:* entity identification has already been performed, so candidate keys of the two heterogeneous relations must already match. Measuring the correspondence score is not necessary. In some circumstances, candidate keys that are matched may appear to be nonequivalent. For example,

consider the case of Name and ID, where all Names are unique. However, such a match implies that every Name maps to a particular ID and vice versa, and therefore one can be substituted for the other in any database transaction in the integrated relation.

2. *One attribute group is assigned CANDIDATE KEY and the other is assigned FOREIGN KEY*: once entity identification has been performed, the candidate keys of the relations to be integrated will be equivalent. Thus, foreign key information adds no value in this case.
3. *Both attribute groups are assigned FOREIGN KEY*: Goodman and Kruskal's Lambda is used to measure the correspondence score in this case. Lambda uses the co-occurrence/non-co-occurrence matrix between instances of the attribute groups to obtain a measure of the proportion of explained variance (i.e., prediction score).

4.2 CODED correspondence score measurement

Two factors should be considered in measuring the correspondence for attribute group pairs having a domain class from the CODED domain hierarchy: (1) the appropriate measurement function to apply to the pair and (2) the magnitude of the correspondence. These two factors are discussed below.

4.2.1 Determination of Measurement Functions

Table 4, an extension of prior work by Sekaran [56] (p. 269), presents the mappings from the domain classes to the functions. Each measurement function is chosen to exploit the domain class properties (e.g., distinctiveness, order) of the attribute groups. Also, each function measures the same concept: the proportion of explained variance between the attribute groups. Thus, the correspondence scores of CODED attribute group pairs can be compared to each other.

For example, the attribute group {Cons-A} is paired with the attribute group {Expt-B}. Our method selects Goodman and Kruskal's Lambda as the appropriate measurement function for this pair. In Table 4, attribute groups assigned the INTERVAL domain class are subcategorized into single (denoted as 1) and multiattribute attribute groups (denoted as N) as the correspondence measurement functions used for single attributes cannot be applied when there are multiple attributes in the attribute group. Correspondence measurement functions for the other domain classes do not have this limitation. Multiattribute attribute groups assigned the DICHOTOMOUS domain class are treated as multiattribute NUMBERS since the DICHOTOMOUS domain class has the limited distance property.

Well-accepted measurement functions are employed in this research [7,29,52], so they will not be discussed further. However, the examples presented in Table 3(a) demonstrates why different functions are necessary in measuring correspondence for different attribute groups, as well as to provide an intuitive understanding of how some of the different functions work.

In Relation A, the attributes Wgt-A and Cons-A have the INTERVAL and CATEGORICAL domain classes, respectively. In Relation B, Kgs-B would have the INTERVAL domain class. To measure the correspondence between Wgt-A and Kgs-B,

the Box-Cox regression function would be used. To measure Cons-A against Kgs-B, an ANOVA would be used.

A regression determines the best fit line through the N -dimensional space containing the attribute values. For example, the points combining Wgt-A and Kgs-B would form a cloud on a two-dimensional space with Wgt-A on one axis and Kgs-B on another. For any functional form (e.g., $Y = \beta X$, $Y = \beta X^2$, $Y = \beta X^3$) there is exactly one line that can be drawn through the points so that the squared distance from the points to the line is minimized. The squared distance is used because points will be normally distributed around a line that genuinely fits the cloud. The slope of this line identifies the optimal mapping between the attribute groups. For example, in the case of Wgt-A and Kgs-B, the regression would determine that 0.45 Wgt-A maps to Kgs-B. The squared distance of the points to the line can be compared to the squared distance of the points to the center of the cloud (i.e., the cloud's mean) to determine the ability of the line to explain the distribution of the points. The explanatory ability of the line is termed the proportion of explained variance and is the correspondence score. In the case of Wgt-A and Kgs-B, this score is 0.992, which suggests that Wgt-A and Kgs-B are very similar (see Table 5).

A regression cannot be used when one of the attributes is CATEGORICAL because CATEGORICAL attributes do not incorporate the notions of order or distance. Instead, an ANOVA is used. In the ANOVA, the values of the INTERVAL attribute are grouped by the values of the CATEGORICAL one. To assess the proportion of explained variance, the distance within the groups is compared to the distance between the groups. For example, within the "Parent" value, the distance between Kgs-B is about 7 kg (i.e., in Table 3(a), the Kgs-B for the tuples with "Parent" ranges from 32 to 39). Between "Parent" and "Child" the distance is 8 kg (i.e., 31 to 39). Because the distances are so close, this suggests that clustering Kgs-B by Cons-A does not significantly reduce the variation in Kgs-B. In fact, in this case the proportion of explained variance is 0.000.

All of the measurement functions described measure the proportion of explained variance between two attribute groups. Since this measure is a cumulative one, it tends to be biased in favor of attribute groups with many attributes. For example, if the proportion of explained variance between the attribute groups Wgt-A and Kgs-B is 0.992, then the proportion of explained variance between {Wgt-A, Scr-A}, and Kgs-B will be at least 0.992. Even a slight artifact association between Scr-A and Kgs-B will cause the measured association to be greater than 0.992.

We adopt the standard procedure for model fitting used in stepwise regression [21] to control this phenomenon. Correspondences that include a multiattribute attribute group are considered only if the correspondence score increase is beyond a certain threshold. The default threshold is set so that the partial contribution of new attributes in the pair is statistically significant at the $\alpha = 0.05$ level.¹ For example, the proportion of explained variance between {Wgt-A, Scr-A}

¹ In statistical language, α is a probability threshold and is compared to p , the actual probability that an event could occur by random chance. $\alpha = 0.05$, the commonly accepted threshold, means we are willing to accept that an association exists because it has only a 5% chance of occurring randomly [31,36].

Table 4. Selection of measurement functions

Group A	Group B						
	NUM(N)	NUM(1)	DATE(N)	DATE(1)	ORD	CAT	DICH
NUM(N)	CC ^d	BT ^b	CC ^d	BT ^b	OL ^e	MA ^h	LO ^j
NUM(1)	BT ^b	BC ^a	BT ^b	BC ^a	OL ^e	AN ^g	PB ^k
DATE(N)	CC ^d	BT ^b	CC ^d	(R ²) ^c	OL ^e	MA ^h	LO ^j
DATE(1)	BT ^b	BC ^a	(R ²) ^c	(R ²) ^c	OL ^e	AN ^g	PB ^k
ORD	OL ^e	OL ^e	OL ^e	OL ^e	ρ^f	λ^i	λ^i
CAT	MA ^h	AN ^g	MA ^h	AN ^g	λ^i	λ^i	λ^i
DICH	LO ^j	PB ^k	LO ^j	PB ^k	λ^i	λ^i	ϕ^l

^a Box-Cox^b Box-Tidwell^c Pearson's Coefficient of Determination^d Canonical Correlation^e Ordered Logit^f Spearman's Coefficient of Rank Determination^g One-Way Analysis of Variance (ANOVA)^h One-Way Multivariate Analysis of Variance (MANOVA)ⁱ Goodman and Kruskal's Lambda^j Logistic Regression/ANOVA/MANOVA^k Point Biserial Correlation Coefficient^l Phi Coefficient**Table 5.** Correspondence score matrix of attribute groups from Relations A and B

Rel. A	Rel. B			
	L-Grd-B	Kgs-B	Expt-B	Pgm-B
Scr-A	0.901	0.209	0.01	0.831
Wgt-A	0.102	0.992	0.240	0.410
Cons-A	0.300	0.000	0.750	0.000
Cm-A	0.705	0.291	0.041	0.792
Mt-A	0.435	0.200	0.224	0.412
{Cm-A, Mt-A}	0.793	0.356	0.469	0.831

and $Kgs-B$ is 0.995. As this value is not significantly higher than 0.992, the two attribute groups are determined not to correspond.

4.3 STRING correspondence score measurement

In many cases, it is not necessary to measure the correspondence score of an attribute group pair with a common STRING domain hierarchy because their domain classes will reveal that they do not correspond. For example, an attribute assigned the ALPHABETIC domain class will not correspond to an attribute assigned the ZERONINE domain class through string matching because the former has no values from 0 . . . 9, while the latter consists mainly of such values. A correspondence score of 0 is automatically assigned to such cases.

If the domain classes reveal nothing about the correspondence, then string distance [59] must be used to measure their correspondence. We adopt the Levenshtein [37] metric to measure string distance between instances. The Levenshtein metric counts the minimum number of insertions, deletions, and substitutions required to transform one string to another. To standardize the Levenshtein metric and map it to a score between 0 and 1, we adopt the formula in Eq. 1, where $StringDistance$ is the Levenshtein metric and $String_A$ and

$String_B$ are the instances being compared. The correspondence score is then the average standardized string distance.

$$1 - \frac{StringDistance}{\max(\text{len}(String_A), \text{len}(String_B))} \quad (1)$$

The accurate measurement of string distance for multiattribute attribute groups requires knowledge of the correct order of the attributes within each group. For example, if the correspondence score between the pair $\{\{Surname, Given_Name\}, \{First_Name, Last_Name\}\}$ were measured, it would differ depending on whether $(Surname, Given_Name)$ was being compared to $(First_Name, Last_Name)$ or $(Last_Name, First_Name)$. The appropriate order can be obtained by measuring the string distance of subsets of such attribute group pairs. For example, if the correspondence score of $\{\{First_Name\}, \{Given_Name\}\}$ were higher than that of $\{\{First_Name\}, \{Surname\}\}$, it would be more appropriate to measure the correspondence score between the groups $(Given_Name, Surname)$ and $(First_Name, Last_Name)$.

5 Matching attribute groups

The third step of the attribute identification process determines the best combination of matching attribute group pairs for attribute integration. At this point, the correspondence scores of all candidate attribute group pairs have been produced. The best combination of matching attribute group pairs cannot be obtained by selecting attribute group pairs with the highest scores because in many cases there will be conflicting matches. For example, consider the matrix of correspondence scores for the elementary school example presented as Table 5. For brevity, only scores for single attributes and one multiattribute attribute group are presented.

As can be seen, the correspondence between $\{Cm-A, Mt-A\}$ and $Pgm-B$ has the same score as the correspondence between $Scr-A$ and $Pgm-B$. Considering these correspondences in isolation, it is not obvious which attribute group $Pgm-B$ should be matched with. However, because $Scr-A$ corresponds more strongly to $L-Grd-B$ and should therefore be matched with it, $Pgm-B$ should be matched with $\{Cm-A, Mt-A\}$. This example illustrates the problem of matching attribute groups. To match one attribute group with another, the matching of all other attribute groups must be considered simultaneously.

The problem of matching attribute groups is a special case of the linear assignment problem: allocate M resources to N

tasks to maximize a cost or benefit function. Only one resource can be allocated to each task. If M and N have different values, either some resource is not allocated or some task does not receive a resource. The linear assignment problem is well studied, and solutions such as the Hungarian algorithm [46] (employed by our method) have been developed. Thus, the attribute group matching task is performed in the following sequence:

1. The correspondence scores of the attribute groups are arranged in three matrices, one for each domain hierarchy.
2. The linear assignment algorithm is applied, and proposed matches are presented to the user.
3. The user accepts or rejects the matches. If the user accepts a match, the matching attribute groups and all attribute groups that are subsets of the matching groups are removed from the matrix. Otherwise, the correspondence score of the match is set to 0. The process then iterates from step 2 until either all attribute groups have been matched or the user halts the process.

The matching task matches all attribute groups from the relation with the fewest attribute groups. However, in many cases, not all attribute groups in a relation match with other attribute groups in another relation. One way of identifying artifact matches is by assessing the correspondence score of the match. Low correspondence scores suggest poor matches. To accommodate this possibility, the user sets a threshold so that matches below this threshold are not presented. The default threshold set is 0.100. This means that less than 10% of the variance between the two attribute groups is explained. The threshold is deliberately set low to minimize the number of genuinely matched attribute groups that are not presented.

6 Validation

Three experiments were conducted to validate the proposed attribute identification method. In the first, the heuristic rules established for attribute classification were tested against several public domain data sets to measure their effectiveness. In the second experiment, the attribute identification method was validated by applying it to integrate attributes from two relations with known confounding characteristics (e.g., synonymity). By applying the method in a controlled environment, we could accurately identify the independent factors that would reduce its efficacy. In the third experiment, our attribute identification method was applied to relations obtained from several online music stores to test its applicability in a real-world setting. These experiments illustrate the viability of the proposed method.

6.1 Experiment 1: Validation of heuristic rules for attribute classification

We compared our domain class assignment process against assignments made by a domain expert on the public domain data sets found in [3, 4, 6, 17, 20, 25, 28, 30, 34]. Most of the data sets were obtained from the CMU and UCI data repositories [5, 13]. Out of 119 attributes, 112 were identified correctly. In most data sets, only one or fewer misclassifications occurred per ten

attributes. Four of the data sets had no misclassification errors. Table 6 summarizes the results. **Data Set** presents a reference for each data set used. **# Attr** counts the number of attributes in the data set. **# Attr \checkmark** indicates the number of attributes identified correctly. The classification performed by the domain expert was taken as a baseline (i.e., the domain expert's classification was assumed correct). **Incorrect Classification** describes which attributes were identified incorrectly, and the nature of the error. Finally, **Reason for Failure** describes why the classification error occurred.

Half of the incorrectly classified attributes were attributes that should have been classified as NUMBER but were classified as ORDINAL. These attributes have the following common properties:

- The data values are represented by **Integers**.
- They have few distinct instances. This property, in combination with the **Integer** data type property, misleads our attribute classification process into classifying these attributes as ORDINAL.
- They are predominantly used to “count” something (e.g., years of Education, No_of_Children). The sole exception was the attribute Time_to_accelerate.

Based on the results of this experiment, we have extended the heuristic rules to capture these properties and correctly classify these attributes to enhance our attribute identification method. New heuristics scan the attribute names for keywords such as “No,” “Cnt,” “Count,” “Year,” or “Yr.”

6.2 Experiment 2: Data with known characteristics

For this experiment, an existing relation of 6090 elementary school students was used (called Relation 1). A duplicate copy of the relation was made (called Relation 2), and the attributes were recoded to simulate a second relation to integrate with the first. To simulate real-world conditions, a scripting program inserted random errors into the relations. For example, in Gender, some males were recorded as females. We conducted this experiment to validate our approach against the following attribute identification problems:

- *Matching attributes have different data types:* in the two relations, Gender and Sex had different data types, as did STLang (student's preferred language) and Pref_Lang (preferred language).
- *Synonyms or homonyms among attributes:* Child in Relation 1 referred to the child's consent form. In Relation 2, it referred to the child's name. Sex and Gender were synonyms, as were StLang and Pref_Lang.
- *Matching attributes have different scales:* Child and Parent consent were more finely detailed versions of Has_Consent (i.e., consent was given only if both the parent and the child consented).
- *An attribute is an abstraction of another:* the country of nationality was a more finely detailed description of the student's race. For example, most Chileans, Brazilians, Argentinians, etc. by birth were coded as Hispanics.
- *Attributes have erroneous or null values:* random errors totaling 10%, 15%, and 20% of all values were introduced into Relation 1 during three successive runs of the experiment.

Table 6. Application of heuristic rules to data sets

Data Set	# Attr	# Attr \checkmark	Incorrect Classification	Reason for Failure
[20]	9	7	No_of_Cylinders and Time_to_accelerate were classified as ORDINAL instead of NUMBER	Too few distinct instances
[3]	11	10	Education was classified as ORDINAL instead of NUMBER	Too few distinct instances
[17]	8	7	Sample_Date was classified as NUMBER instead of DATE	Day values of the attribute were given the value 00
[30]	33	33	No error	No error
[4]	6	6	No error	No error
[28]	6	6	No error	No error
[6]	15	14	Education was classified as CATEGORICAL instead of ORDINAL	Instances of Education were found in the dictionary
[34]	27	25	#_in_family and #_kids classified as ORDINAL instead of NUMBER	Too few distinct instances
[25]	4	4	No error	

- *Attribute group matchings could be identified:* the three attributes Last, First, and MI together formed Child. Child and Parent together were the same as Has_Consent.

To validate the robustness of the method, we inserted 609, 915, and 1218 errors (i.e., 10%, 15%, and 20% of the data, respectively) into the duplicate relation for every attribute. Note that these error levels are extreme. While it is likely that 10% of the tuples in any relation may have errors, data quality must be very poor before 10% of the attribute values contain error. Results with a score of less than 0.1 were considered noninteresting. The final matching results for these experiments are presented in Tables 7, 8, and 9, respectively, sorted in order of measure type and then score. The tables are interpreted as follows. **No** is a reference number for that pairing in the experiment. **Attr 1** is the attribute group from Relation 1. **Attr 2** is the attribute group from Relation 2 that matches **Attr 1**. **DH** is the domain hierarchy used to select the correspondence score measurement function. **S** indicates string matching. **C** (for Coded) indicates a statistical function. **Result** indicates whether the pairing was correct. **Partial** means that a subset of two matching attribute groups was identified. **Remarks** describes matching pairs that were dropped because they conflict with matching pairs with a higher correspondence score.

The results of this experiment show how robust our method is subject to data error. At 10% error, all of the correct matching attribute group pairs were correctly presented. At 15% error, one incorrect matching pair was presented. Also, as a result of that incorrect matching pair, two correct matching pairs were not presented. A second round of matching would identify these remaining correct pairs. At 20% error, one correct matching pair would never be detected, the pairing of {Parent, Child} with Has_Consent. Note that at 20% error, fully 20% of the attribute values of Parent and 20% of the attribute values of Child contained errors.

This experiment suggests that our method can work for cases where synonyms, homonyms, attributes with different scales, attribute groups, data abstractions, and erroneous or null values occur between relations to be integrated. Furthermore, it provides us with an assessment of the method's robustness in handling erroneous data instances by demonstrat-

ing that (at least for one case) the method fails only when a large percentage of data is erroneous.

6.3 Experiment 3: Validation on a real-world setting

To demonstrate that our attribute identification method is applicable and useful in a real-world setting, we used it to integrate five relations extracted from the online music stores CDWorld [10], MassMusic [47], MusicForce [51], Riddim [53], and X-Radio [60]. This data was HTML text data, so there was little available schema information that could be analyzed automatically. Hence the integration of these five relations provided an ideal real-world test for the applicability of the method. Furthermore, because the "implicit" schema of the five relations was highly similar, it was possible for us to verify the results. Attribute identification was performed on five relations to afford us sufficient replication to evaluate the generalizability of the results. With five relations we could test our method on up to 32 (i.e., 2^5) pairs of data sets.

Table 10 presents a summary of the five relations. In most of the relations, "artist" referred to the recording artist or group, "title" to the CD title, "label" to a production code, "No" to a serial number found on the CD, "date" to the date of release, and "price" to the sale price of the CD. However, there were exceptions. For CD World, "price" referred to the CD list price, while "Buy CD" referred to the market price. Furthermore, for this relation, "label" referred to the recording studio. For Riddim and X-Radio, "Maker" referred to the recording studio.

Entity identification was performed by comparing the Title and Artist attributes of the relations. No matching entities were found for three pairs of relations – Riddim vs. Musicforce, Musicforce vs. X-Radio, and Massmusic vs. MusicForce. Attribute identification was performed on the attributes of the relation-pairs with successful entity identification. The mean and median success rates for attribute matching averaged 0.719 and 0.833 (i.e., on average our method successfully identified 72% of matched attributes).

All matching errors occurred with CODED attribute groups, and were caused by one of two factors:

Table 7. Results on transformed relations (10% error)

No	Attr 1	Attr 2	DH	Score	Result	Remarks
1	{First, MI, Last}	Child	S	0.907	Correct	
2	Gender	Sex	C	0.802	Correct	
3	StLang	Pref_Lang	C	0.725	Correct	
4	{Child, Parent}	Has_Consent	C	0.653	Correct	
5	Ethnic	Nation	C	0.576	Correct	
6	Child	{Has_Consent, Nation}	C	0.154	Incorrect	Dropped because of 4

Table 8. Results on transformed relations (15% error)

No	Attr 1	Attr 2	DH	Score	Result	Remarks
1	{First, MI, Last}	Child	S	0.900	Correct	
2	Gender	Sex	C	0.732	Correct	
3	StLang	Pref_Lang	C	0.651	Correct	
4	{Ethnic, Parent}	{Has_Consent, Nation}	C	0.469	Incorrect	
5	{Child, Parent}	Has_Consent	C	0.382	Correct	Dropped because of 4
6	Ethnic	Nation	C	0.378	Correct	Dropped because of 4
7	{Gender, Parent}	{Has_Consent, Pref_Lang}	C	0.354	Incorrect	Dropped because of 2

Table 9. Results on transformed relations (20% error)

No	Attr 1	Attr 2	DH	Score	Result	Remarks
1	{First, MI, Last}	Child	S	0.893	Correct	
2	Gender	Sex	C	0.631	Correct	
3	StLang	Pref_Lang	C	0.536	Correct	
4	{Ethnic, Parent}	{Has_Consent, Pref_Lang}	C	0.407	Incorrect	
5	{ST_Lang, Parent}	{Has_Consent, Nation}	C	0.356	Incorrect	
6	Parent	Has_Consent	C	0.347	Partial	Dropped because of 4 and 5
7	Ethnic	Nation	C	0.311	Correct	Dropped because of 4 and 5
8	{Ethnic, Child}	{Has_Consent, Sex}	C	0.264	Incorrect	Dropped because of 4, 5, 7

- **Attribute groups were matched when no corresponding attribute group existed.** In all of these cases, the correspondence scores of the matches were very low. For example, `MassMusic.Date` was matched with `Riddim.No` even though the correspondence score was 0.048. Therefore, we suggest that attribute groups matched with correspondence scores lower than 0.100 should be considered as having no matching attribute groups in the corresponding relation. This value of 0.100 should be decreased if the number of matching tuples is very low (e.g., less than 100).
- **Data sets to be integrated came from organizations with different business models.** We observed that it was particularly difficult to integrate the `Riddim` and `X-Radio` data sets (both owned by Radix International) with those of other organizations. An examination of our results suggests that attribute identification failed due to the different marketing behavior of Radix International. For example, most correspondences between the `Price` attributes of Radix's data sets and other vendors tended to have low scores. This suggests that Radix has a different pricing scheme than other vendors. Also, the highest error rates occurred when integrating the `Riddim` data set with that of other vendors. `Riddim` is a branch of Radix International that specializes in reggae music. We suspect that their specialization enables them to price their niche product at a more competitive level than "normal" CD retailers such as `CDWorld`.

This result suggests that our method is useful. In many cases, database integration specialists using purely domain knowledge are likely to have matched the `Price` attributes of the five stores. Our method, however, successfully detected that the `Prices` of the stores did *not* match.

6.4 Implication and discussion

Overall these experiments have demonstrated that the proposed method is feasible, applicable, and robust for attribute identification. The first experiment demonstrated that the heuristic rules could be employed to correctly classify attributes and group attributes.

The second experiment demonstrated the method's ability to function in situations where schema information was insufficient for attribute identification. Specifically, it could correctly group attributes even when they had different names, data types, scales, and levels of abstraction. Also, this experiment demonstrated the robustness of the method. When 10% of the data values were erroneous or missing, the method could still perform well. Note that it is very rare for 10% of data values to be erroneous or missing.

The final experiment determined the applicability of the method. The method successfully identified a case where a human being would have erroneously identified two attributes as matching. In an integrated database, `Riddim`'s price attribute

Table 10. Real-world relations employed in experiment

Name	Citation	No. Tuples	Attributes
CDWorld	[10]	108105	Artist, Title, Label, No, Date, Price, Buy CD
MassMusic	[47]	41412	Artist, Title, Label, Date, Price
MusicForce	[51]	2991	Artist, Title, Price
Riddim	[53]	7502	Artist, Title, Label, No., Maker, Price
X-Radio	[60]	1262	Artist, Title, Label, No., Maker, Price

should be kept separate, as the pricing strategy is distinct from that of the other companies.

7 Summary and future research

An attribute identification method has been presented that assumes the successful performance of entity identification. This method examines the schema and attribute values to identify appropriate functions for measuring correspondence between the attribute groups. The correspondence scores are then used to identify matching attribute groups. One-to-one matching between attribute groups are established to avoid inferring redundant attributes. By applying the proposed method, the database integration specialist plays a supporting role for attribute identification, mainly in validating the results of each step.

7.1 Limitations

The method is designed to supplement existing research by enabling attribute identification in situations where schema and summary instance information prove insufficient. The results from empirical analysis suggests that the method performs correctly for its intended purpose, i.e., identifying attributes for two relations when entity identification has been performed. Nevertheless, the approach has limitations. The most obvious limitation is that it cannot be employed without a prior successful entity identification.

Second, it is possible that, when our method is employed simultaneously with a method employing only schema and summary instance information, both methods produce incompatible results. In such situations, it is possible that our method performs incorrectly, especially given the limitations outlined below.

Our heuristics (Sect. 3) have only been applied for the data sets presented in Sect. 6. Although every attempt was made to test the method, there is no feasible way to obtain a random sample of data sets. As a result, we cannot guarantee that the method is applicable to all domains [16]. Furthermore, our experiments identified some kinds of attributes that were not successfully identified. However, given that the data sets were obtained from multiple sources, some degree of orthogonality can be assumed. This assures that our method has wide applicability [2].

The applicability of the correspondence score measurement functions (Sect. 4) is likely to be robust. All of these functions are in wide use, especially in the soft sciences (e.g., psychology, sociology, economics, medicine), and have been shown to be reliable in situations where data instabilities such as measurement error are prevalent [29]. In fact, in many of

these soft sciences, error rates are substantially higher than the error rates tested here [14, 16]. Nevertheless, results from these studies are often considered satisfactory for enacting social policy (i.e., for decisions of greater consequence than attribute identification) precisely because the robustness of these measures is accepted. Furthermore, the information provided by the correspondence score measurement functions is in addition to existing information provided by the database schemas. Thus, the pool of information is increased, and attribute identification is generally improved. However, if these data instabilities are unusually large, the method fails. This was demonstrated in Sect. 6.2.

One-to-one matching (Sect. 5) has perhaps the strongest set of assumptions. One strong assumption is that every attribute group in one relation has at most one match in another relation. In cases where a relation has redundant attributes (e.g., calculated attributes), this is not the case. Section 6.3 presents one instance of a redundant attribute. In the CDWorld data set, `Buy CD` and `Price` were redundant. `Buy CD` was effectively `Price` with a fixed discount. This issue was handled successfully because neither `Buy CD` nor `Price` correlated with any of the other attributes. There is no guarantee in other data sets that this would be the case.

It is for these reasons that in our system, the database integration specialist is always given the power to override a classification made by the method. The human's ability to adapt to changing circumstances complements the method's ability to tirelessly analyze large amounts of data (i.e., the instances). However, the human's decision is informed by the additional information (e.g., domain classes and correspondence scores) provided by our method.

7.2 Future research

This research can be extended in three directions. First, we are reviewing other relevant measurement functions for adoption in our attribute identification method. In many cases, due to limitations of the functions used, correspondence between attribute groups is not measured correctly. For example, if an attribute group pair with a common `NUMBER` domain class corresponds through the function $\sin(X) = Y$, our attribute identification method is likely to determine that the attribute group pair has no correspondence. However, functions that can measure correspondence for attribute groups with a wide range of characteristics tend to have poor accuracy. Second, we are investigating the use of the domain class assignment and the correspondence score measurement processes described in this paper to improve the knowledge discovery in databases (KDD) process. These correspondence scores can be used to find attributes that will be susceptible to mining with clustering [23] or association rule algorithms [1]. Finally, we intend

Table 11. Initially generated hypotheses

Type	NUM	DATE	ORD	CAT	DICH
Integer	✓	✓	✓	✓	✓
Decimal	✓	✓			
String	✓	✓	✓	✓	✓
Date		✓			

to conduct more experiments to validate the method. Experiments to compare the method's performance as a stand-alone system with the method's ability to complement a domain expert are planned.

Acknowledgements. We would like to thank Lin Yong for his work in extracting the online music store data, Dr. Veda Storey, and Dr. Marianne Winslett for advice and comments, Dr. Amit Das, Dr. Michael Li, the UCLA Statistics Department, especially Dr. Jan de Leeuw, and Dr. Richard Berk, Dr. Henry Rubin, and the sci.stat.consult newsgroup community for their invaluable advice regarding the statistics in this paper, Dr. Marti Singer and Lydia Williams for helpful editorial comments, and the multitude of people whose suggestions in one way or other have shaped this article.

We would also like to thank associate editor Louiqa Raschid, and two anonymous reviewers for helpful comments and insights.

A Domain assignment and measurement function selection rules

In this appendix, we present a justification for the various rules and heuristics used to assign the domain classes and measure the correspondence scores of the attribute groups. Wherever possible, the examples provided in Table 2 are used to illustrate the rules discussed.

A.1 Assignment of CODED domain classes

Attributes with a particular data type can have many possible CODED domain classes. For example, an attribute with the **Integer** data type can be assigned the **NUMBER** domain class (e.g., `Salary`), the **DATE** domain class (e.g., Julian dates), the **ORDINAL** domain class (e.g., `Pgm-B` where "1" is "Gifted," "2" is "Normal," etc.), the **CATEGORICAL** domain class (e.g., `Cons-A`, where "1" is "Both," "2" is "Parent," etc.), or the **DICHOTOMOUS** domain class (e.g., `Expt-B` where "1" is "Yes"). Table 11 describes the possible CODED domain classes for each data type. A set of heuristic rules is established to determine the correct CODED domain class. These heuristic rules are extensions of work done in [12,55]. Where rules are direct translations of those presented in [55], the rules are cited.

The data type is the principal kind of schema information employed to determine the domain classes. Different database management systems (DBMSs) use different names to describe the same data types. In this research, we consider the four most commonly used data types for business data: **Integer**, **Decimal**, **Date**, and **String**. Data types such as **Varchar**, **Memo**, **Currency**, and **Boolean** are considered as **String**, **String**, **Decimal**, and **Integer**, respectively. Some DBMSs employ other data types such as **Graphic**, **OLE_object**, etc. that require graphic, document matching or other algorithms to

integrate. This research does not attempt to integrate attributes with those data types.

These rules are established to assign domain classes in the sequence of **DICHOTOMOUS**, **DATE**, **ORDINAL**, **CATEGORICAL**, and **NUMBER**. This sequence has been established based on the level of difficulty in assigning domain classes. Once a domain class is identified, the assignment process terminates, as each attribute may have only one domain class per domain hierarchy.

Dichotomous: one rule is sufficient for determining the **DICHOTOMOUS** domain class.

1. If the data type of the attribute is **Integer** or **String**, and the number of distinct attribute values is one or two, the attribute is assigned the **DICHOTOMOUS** domain class.

Justification: **DICHOTOMOUS** attributes are defined as having only one or two distinct instances.

Date: if an attribute has more than two distinct instances and satisfies one of the following rules, it is assigned the **DATE** domain class:

1. The Attribute has the **Date** data type.
2. A date-specific function (e.g., `Month()`, `Day_of.Week()`, etc.) is applied to the attribute by an application in the database system. Many DBMS systems incorporate interfaces to programming languages such as C or Visual Basic. All application source code is searched for reserved words associated with an attribute. For example, if the function `Month(Nomroll.DOB)` (i.e., *date of birth*) is found in the application source code, there is strong evidence that `Nomroll.DOB` is a **DATE**.
3. The character strings "January," "February," etc. are found in all instances.
4. The attribute values have a consistent format that conforms to a standard date format. For example, `DD/MM/YY`, `DDD/YY`, or Julian dates.

Justification: these rules analyze the attribute values against almost all well-known date formats.

Ordinal and categorical: if neither **DATE** nor **DICHOTOMOUS** can be assigned to an attribute, it is considered for the **ORDINAL** or **CATEGORICAL** domain class. An attribute can have one of these two domain classes if it meets at least one of the following conditions:

1. The attribute has a data length of less than 10 with less than 26 distinct values. The values 10 and 26 are defaults and can be adjusted by the database integration specialist. *Justification:* this rule is adopted from the research done in SNOUT [55]. The value 10 is chosen as the threshold to support coding schemes such as the military alphabet (where a code of "W" is "Waterfall"). The value 26 is chosen as a threshold because many coding schemes use letters of the English alphabet.
2. The attribute is a foreign key and is the candidate key of a small relation (fewer than 26 instances) or system relation. *Justification:* this rule attempts to capture **CATEGORICAL** or **ORDINAL** attributes that are presented as codes such as

Pgm-B. Such codes are normally defined in a separate supporting relation (called a system table). The value 26 is a default suggested by [55].

3. During data entry the attribute values are selected from a list (e.g., pop-up window). Selection from a pop-up window can be discovered by searching for pop-up-related reserved words in the application source code.
Justification: selection implies choice from a set of ordered or unordered categories. Depending on the way in which the source code was written, this rule may or may not extract useful information.

The following four rules are applied to determine whether the attribute should be assigned either the ORDINAL or CATEGORICAL domain class.

1. If an attribute has the Integer data type, and has a length greater than 2 (the size required for 25 distinct instances), it is assigned the ORDINAL domain class.
Justification: this rule captures ORDINAL attributes that have character positions with independent meaning. For example, the Mercedes-Benz E series is structured on a three-digit code, where the first digit refers to the size of the car, and the remaining digits refer to the model.
2. If the values of the attribute can be meaningfully ranked, then the attribute is assigned the ORDINAL domain class. For example, the values of the attribute L-Grd-B can be ranked as (A, B, C, D, F). We consider an attribute meaningfully ranked if it meets one of the following criteria:
 - In the change history, most (> 90%) of the attribute values are updated in the same way.
Justification: a consistent update pattern indicates ordering.
 - The attribute values of the strings contain nonnumeric characters. When alphabetically sorted, the leftmost character of each value is at most two characters less than the next attribute value.
Justification: this allows us to identify ORDINAL attributes such as L-Grd-B (i.e., values “A,” “B,” “C,” “D,” “F”) and Signal_Codes (i.e., values “Able,” “Baker,” “Charlie,” “Delta,” “Echo,” etc.).
3. If all values of an attribute can be found in a dictionary or spell-checking reference, then the attribute is assigned the CATEGORICAL domain class. For example, all Occupations (e.g., “Engineer,” “Accountant,” “Manager”) will be found in a dictionary.
Justification: the meaning of each word in a dictionary can be considered in isolation from the other words. Thus, ordering is not relevant. Also, the only implementable sort order that can be established on words is alphabetic, which often has no semantic value. Attributes that are meaningful when sorted alphabetically are captured by Rule 2.
4. If all distinct instances of the attribute except one can be arranged in running order, and the exception ends with a 9, then the attribute is assigned the CATEGORICAL domain class. For example, an attribute with instances {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 99} would be assigned this domain class.
Justification: this rule exploits a “hack” employed by many database administrators and data analysts who represent special values such as “null” with a numeric string terminated by 9.

Number: Finally, the NUMBER domain class can be assigned to attributes as long as any of the following rules are satisfied:

1. The instances of the attribute are always displayed with a measurement symbol (e.g., \$1200.00, 25°C, 35 μm.).
Justification: most of the metrics with measurement symbols are on the interval measurement scale. Differences in the structure of the source code will affect the success of this rule.
2. The last character of every instance has a value of 0 or 5.
Justification: This rule captures the tendency to “round off” the values of attributes with NUMBER domain classes. For example, one is more likely to be paid a Salary of \$15,000 than \$15,263.
3. The attribute has the Integer or Float data type, is not a candidate or foreign key, and is never displayed on a report or screen in a nonnumerical format.
Justification: this rule attempts to eliminate candidate key attributes, which are often not appropriate for data analysis.
4. The attribute has the String data type, contains only numeric and associated characters (e.g., ‘.’, ‘-’), is not a candidate or foreign key, and is never displayed in a report or screen in a nonnumerical format.
Justification: attributes with the String data type that contain nonnumeric values cannot be manipulated algebraically.

After the domain classes have been assigned, the database integration specialist may review the results and change any inappropriate assignment.

A.2 Attribute groups with CODED domain classes

In this section, the justification for each assignment rule established and presented in Table 3 is discussed.

1. An attribute assigned the NUMBER domain class combined with an attribute assigned the DATE domain class results in the DATE domain class for the combined attribute group. This rule reflects situations when numbers are used to increment or decrement a date.
2. An attribute assigned the ORDINAL domain class combined with an attribute assigned the ORDINAL domain class results in the CATEGORICAL domain class for the combined attribute group. Intuitively, when two attributes that are assigned the ORDINAL domain class are combined, the resultant attribute group should be assigned the ORDINAL domain class. This is not allowed because the ordering priority is user determined and not an inherent characteristic of attributes. Since the order property of the attributes cannot be used, only the distinctness property is applicable.
3. When combining an attribute assigned an INTERVAL (i.e., NUMBER or DATE) domain class with an attribute assigned a CATEGORICAL domain class, a domain class will not be assigned to the combined attribute group. Attributes with these two domain classes cannot be combined meaningfully. For example, if Scr-A and Cons-A were combined, the distance property of Scr-A would be lost. Also, since small differences in salary are not meaningful, the distinctness property is no longer applicable.

4. When combining an attribute assigned an INTERVAL domain class with an attribute assigned an ORDINAL domain class, a domain class will not be assigned to the combined attribute group. The reasons for this are similar to that of rules 2 and 3.
5. When combining an attribute assigned an INTERVAL domain class with an attribute assigned a DICHOTOMOUS domain class, the interval domain class is assigned to the attribute group. DICHOTOMOUS attributes are different from other NOMINAL attributes since they can be treated as if the distance property applied to them. Thus, they can be analyzed as INTERVAL attributes.
6. When two DICHOTOMOUS attributes are combined, the resultant domain class is DICHOTOMOUS. Note that, in this case, DICHOTOMOUS refers to a set of DICHOTOMOUS attributes.

A.3 Functions selection rationale for CODED attribute groups

Table 4 presented the functions used for measuring the correspondence between attribute groups with CODED domain classes. In this section, the rationale for applying the functions specified in this table is elaborated.

- **Linear regression functions:** an attribute group pair assigned the DATE domain class cannot be multiplied or divided. Thus, the correspondence of any pair with the DATE domain class will always be linear with respect to the attribute groups. Consider the pair of attribute groups $\{\text{Project_Start}, \text{Project_Initiated_On}\}$. The linear function that correlates them will be of the form $\text{Project_Start} + C = \text{Project_Initiated_On}$, and not of the form $\alpha \times \text{Project_Start}^\beta + C = \text{Project_Initiated_On}$. The Pearson's Coefficient of Determination and Canonical Correlation are appropriate for measuring their correspondence score. The result extracted for analysis is R^2 . A good discussion of these measurement functions can be found in [52].
- **Robust regression functions:** an attribute group assigned the NUMBER domain class may correspond to an attribute group assigned an INTERVAL domain class in a nonlinear fashion. We select the Box-Tidwell and Box-Cox functions to measure the correspondence score, as of all the appropriate functions, these two provide the best tradeoff between computation speed and accuracy. Canonical Correlation is used in the case of two multiattribute attribute groups with NUMBER domain classes because there is no equivalent to the Box-Cox function that handles such a case. The Box-Cox and Box-Tidwell functions are commonly discussed in statistics textbooks (e.g., [21,26]). R^2 is the result derived from these functions.
- **Ordinal functions:** the Ordered Logit and Spearman's Rho (squared) are the only two functions we discovered that exploit order information without exploiting distance information. The use of Spearman's Rho is discussed in most introductory statistics textbooks (e.g., [7,31]). The semantic equivalence of Spearman's Rho and R is discussed in [32]. A good discussion of the Ordered Logit and converting the results of the Ordered Logit to R^2 can be found in [44].
- **MANOVA and ANOVA:** these functions are used when one attribute group is assigned the CATEGORICAL domain class while the other is assigned the INTERVAL domain class. The MANOVA is a generalization of the ANOVA used when the attribute group on the interval scale has multiple attributes. The η^2 value of the MANOVA and ANOVA has the same semantics as the regression coefficient R^2 measured by the various regression functions [31]. Both of these functions are found in most textbooks on regression and linear modeling (e.g., [52]).
- **Goodman and Kruskal's Lambda:** this function transforms the result of the chi-square test into a number that is comparable to η^2 and R^2 . Lambda is also used when one attribute group has the ORDINAL domain class, as we have found no function that exploits ordering information in that situation. Lambda is thoroughly discussed in [22].
- **Logistic Regression, Point Biserial Correlation, Phi Coefficient:** all of these measurement functions exploit the pseudodistance information found in attribute groups with the DICHOTOMOUS domain class. A good discussion of these functions can be found in [7,44].
- **The relations to be integrated are preferably in 3NF:** relations in normal forms lower than 3NF have nonkey functional dependencies that can cause errors during the attribute matching step. The following kinds of dependencies will cause errors:
 - **Repeating groups:** our method will not be able to identify the multiple matches of repeating groups found in non-First Normal Form (1NF) relations. For example, in the integration of `Student_Course(Matric_No, Course_1_Grade, Course_2_Grade)`, and `Student_Course(Matric_No, Course_Grade, Course_Grade)`, `Course_Grade` matches with `Course_1_Grade` and `Course_2_Grade` separately. However, our method will only be able to match `Course_Grade` with one of the non-1NF attributes.
 - **One-to-one transitive dependencies:** if a relation in Second Normal Form (2NF) has a one-to-one transitive dependency, then the attributes with the dependency are synonyms (i.e., synonyms *within* the relation). Synonyms within a relation are redundant. For example, in the relation `Student(Matric_No, Dept_Code, Dept_Name)`, `Dept_Code` and `Dept_Name` are synonyms. As in the non-1NF case, our method will only successfully match one of the two synonymous attributes. If only one-to-many transitive dependencies exist, matching attributes and attribute groups will be successfully detected.

References

1. Aggarwal CC, Yu PS (1998) Mining large itemsets for association rules. *Bulletin of the IEEE Computer Society technical committee on data engineering*, 2(1):23–31
2. Bekker PA, Merckens A, Wansbeek TJ (1994) Identification, equivalent models, and computer algebra. Academic Press, New York
3. Berndt ER (1991) Determinants of wages from the 1985 current population survey. In: *The practice of econometrics: classic and*

- contemporary, Chap. 5, Addison-Wesley, Reading, MA, pp 193–209 [online] <http://www.stat.cmu.edu/datasets/>.
4. Biblarz TJ, Raftery AE (1993) The effects of family disruption on social mobility. *Am Sociol Rev* 58:97–109
 5. Blake C, Keogh E, Merz CJ (1998) UCI repository of machine learning databases. [online] <http://www.ics.uci.edu/~mllearn/MLRepository.html>
 6. Bureau of Labor Statistics (1995) March 1995 population survey - classical families. [online] <http://www.stat.ucla.edu/data/fpp>, 1995
 7. Burns RB (1997) Introduction to research methods, 3rd edn. Addison-Wesley, Reading, MA
 8. Castano S, De Antonellis V (1999) A schema analysis and reconciliation tool environment for heterogeneous databases. In: Abstracts of the international database engineering and applications symposium, Montreal, pp 53–62
 9. Castano S, De Antonellis V, Fugini M, Pernici B (1998) Conceptual schema analysis: techniques and applications. *ACM Transactions on database systems*, 23(3):286–333
 10. Cdworld (1999) The largest internet discount entertainment store. [online] <http://www.cdworld.com/>
 11. Chiang RHL, Baron TM, Storey VC (1996) A framework for the design and evaluation of reverse engineering methods for relational databases. *Data Knowledge Eng* 21(1):57–77
 12. Chua C, Chiang RHL, Lim E-P (1998) A heuristic method for correlating attribute group pairs in data mining. In: Abstracts of the international workshop on data warehousing and data mining, (DWDM'98), Singapore, pp 29–40
 13. StatLib (1998) [online] <http://www.stat.cmu.edu/>
 14. Cohen J (1988) Statistical power analysis for the behavioral sciences, 2nd edn. Erlbaum, Mahwah, NJ
 15. Cohen WW (1998) Integration of heterogeneous databases without common domains using queries based on textual similarity. In: ACM SIGMOD conference on management of data, Seattle, WA, pp 201–212
 16. Cook TD, Campbell DT (1979) Quasi-experimentation: design and analysis issues for field settings. Houghton Mifflin, Boston, MA
 17. Cox LH, Johnson MM, Kafadar K (1982) Exposition of statistical graphics technology. In: Abstracts of ASA statistical computation section, Cincinnati, OH, pp 55–56
 18. Dao SK, Perry B (1995) Applying a data miner to heterogeneous schema integration. In: The first international conference on knowledge discovery and data mining, Montreal, pp 63–101
 19. Dey D, Sarkar S, De P (1998) A probabilistic decision model for entity matching in heterogeneous databases. *Manage Sci* 44(10):1379–1395
 20. Donoho D, Ramos E (1982) PRIMDATA: data sets for use with PRIM-H. [online] <http://www.stat.cmu.edu/datasets/>
 21. Draper NR, Smith H (1981) Applied regression analysis. Wiley, New York
 22. Everitt BS (1977) The analysis of contingency tables. Chapman & Hall, London
 23. Everitt BS (1980) Cluster analysis. Heinemann, Portsmouth, NH
 24. Fadous R, Forsyth J (1975) Finding candidate keys for relational data bases. In: Abstracts of the ACM-SIGMOD international conference on management of data, San Jose, CA, pp 204–210
 25. Fienberg SE, Makov UE, Sanil AP (1994) A bayesian approach to data disclosure: optimal intruder behavior for continuous data. Technical report 11/94, Carnegie-Mellon University, Pittsburgh, PA
 26. John Fox (1997) Applied regression analysis, linear models, and related methods. Sage Publications, Thousand Oaks, CA
 27. Ganesh M, Srivastava J, Richardson T (1996) Mining entity-identification rules for database integration. In: Abstracts of the second international conference on knowledge discovery and data mining (KDD-96), Portland, OR, pp 291–294
 28. Greaney V, Kelleghan T (1984) Equality of opportunity in Irish schools. Educational Company, Dublin
 29. Hair Jr JF, Anderson RE, Tatham RL, Black WC (1998) Multivariate data analysis with readings, 5th edn. Prentice-Hall, New York
 30. Heston A, Summers R (1991) The penn world table (mark 5): an expanded set of international comparisons, 1950–1988. *Q J Econom* 8(6):327–368
 31. Jaccard J, Becker MA (1990) Statistics for the behavioral sciences, 2nd edn. Wadsworth, Boston, MA
 32. Kendall M, Gibbons JD (1990) Rank correlation methods, 5th edn. Oxford University Press, Oxford
 33. Kim W, Seo J (1991) Classifying schematic and data heterogeneity in multidatabase systems. *IEEE Comput* 24(12):12–18
 34. Kohavi R (1996) Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In: Abstracts of the second international conference on knowledge discovery and data mining, Portland, OR, pp 202–207
 35. Larson JA, Navathe SB, Elmasri R (1989) A theory of attribute equivalence in databases with application to schema integration. *IEEE Transactions on software engineering*, 15(4):449–463
 36. Lehman RS (1988) Statistics and research design in the behavioral sciences. Wadsworth, Boston, MA
 37. Levenshtein VI (1966) Binary codes capable of correcting deletions, insertions and reversals. *Cybernet Control Theory* 10(8):707–710
 38. Li W-S, Clifton C (1993) Using field specifications to determine attribute equivalence in heterogeneous databases. In: Abstracts of the third international workshop on research issues on data engineering: interoperability in multidatabase systems, Vienna, pp 174–177
 39. Li W-S, Clifton C (1994) Semantic integration in heterogeneous databases. In: Abstracts of the 20th VLDB conference, Santiago, Chile, pp 1–12
 40. Li W-S, Clifton C (2000) SEMINT: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data Knowledge Eng* 33(1):49–84
 41. Lim E-P, Chiang RHL, Cao Y-Y (1999) Tuple source relational model: a source-aware data model for multidatabases. *Data Knowledge Eng* 29(1):83–114
 42. Lim E-P, Srivastava J (1993) Entity identification in database integration: an evidential reasoning approach. In: Abstracts from the international symposium on next generation database systems and their applications, Fukuoka, Japan, pp 151–158
 43. Lim E-P, Srivastava J, Prabhakar S, Richardson J (1993) Entity identification in database integration. In: Abstracts of the ninth international conference on data engineering, Vienna, pp 294–301
 44. Long JS (1997) Regression models for categorical and limited dependent variables. Sage Publications, Thousand Oaks, CA
 45. Lucchesi CL, Osborn SL (1978) Candidate keys for relations. *J Comput Sys Sci* 17(2):270–279
 46. Martello S, Toth P (1987) Linear assignment problems. *Ann Discrete Math* 31:259–282
 47. Mass music-earth's largest music store (1999) [online] <http://massmusic.com/cgi-bin/mr.cgi?page=index.html>
 48. Miller GA, Beckwith R, Fellbaum C, Gross D, Miller KJ (1990) Introduction to wordnet: an on-line lexical database. *Int J Lexicog* 3(4):235–244

49. Mirbel I (1997) Semantic integration of conceptual schemas. *Data Knowledge Eng* 21(2):183–195
50. Monge AE, Elkan CP (1996) The field matching problem: algorithms and applications. In: Abstracts of the second international conference on knowledge and data mining, Portland, OR, pp 267–270
51. Musicforce.com | welcome (1999) [online] <http://www.musicforce.com/>
52. Neter J, Wasserman W, Kutner MH (1989) Applied linear regression models, 2nd edn. Irwin, Homewood, IL
53. Riddim music (1999) [online] <http://www.riddim.com/>
54. Scheuermann P, Li W-S, Clifton C (1998) Multidatabase query processing in global keys and attribute values. *J Am Soc Inform Sci* 49(3):283–301
55. Scott PD, Coxon APM, Hobbs MH, Williams RJ (1997) SNOUT: an intelligent assistant for exploratory data analysis. In: Abstracts from first European symposium on principles of data mining and knowledge discovery PKDD '97, Trondheim, Norway, pp 189–199
56. Sekaran U (1992) Research methods for business: a skills building approach. Wiley, New York
57. Sheth AP, Gala SK (1989) Attribute relationships: an impediment in automating schema integration. In: Abstracts from the workshop on heterogeneous database systems, Evanston, IL, pp 1–7
58. Singh MP, Cannata PE, Huhns MN, Jacobs N, et al (1997) The Carnot heterogeneous database project: implemented applications. *Distributed Parallel Databases J* 5(2):207–225
59. Stephen GA (1992) String search. Technical report TR-92-gas-01, School of Electronic Engineering Science, University College of North Wales
60. X-radio.com: The internet's number one electronic music store (1999) [online] <http://www.x-radio.com/>
61. Yu CT, Jia B, Sun W, Dao SK (1991) Determining relationships among names in heterogeneous databases. *ACM SIGMOD Record* 20(4):79–80
62. Yu CT, Sun W, Dao S, Keirse D (1990) Determining relationships among attributes for interoperability of multi-database systems. In: Abstracts of the workshop on multi-database and semantic interoperability, Tulsa, OK, pp 251–257
63. Zhao JL (1997) Schema coordination in federated database management: a comparison with schema integration. *Decision Support Sys* 20(3):243–257